Comparative Analysis of Image Colorization Techniques

Group 15
Link to our repo: https://github.com/Karl1018/final_project_3dcv

Yuchen Li
Data and Computer Science
3759412
yuchen.li@stud.uni-heidelberg.de
GAN model, Experiment design, Demo

Maiqi Zhou
Data and Computer Science
4732289
maiqi.zhou@stud.uni-heidelberg.de
Diffusion model, Detection and dataset of repetitive objects

Xichu Yu
Data and Computer Science
4732331
xichu.yu@stud.uni-heidelberg.de
CNN model, Evaluation

Abstract

Automatic image colorization has made significant advancements with the advent of deep learning techniques. Against this background, this project provides a comprehensive comparison of three cutting-edge image colorization methods: Convolutional Neural Networks (CNNs), Generative Adversarial Networks (GANs), and Diffusion Models. We implemented the three mentioned models and delved into the foundational principles of each method, outlining our implementational decisions. We also designed experiments and leveraged different evaluation metrics to support our analysis. Furthermore, this report discusses our research effort for the formal project proposal from the professor, the coloration of repetitive objects, and explains why it was ultimately abandoned.

1. Introduction

In the field of digital image processing, automatic image colorization emerges as a vital technique. The process of colorizing grayscale images involves artificially adding color to monochrome pictures, which historically required skilled artists and extensive manual effort. However, the evolution of deep learning has revolutionized this task, enabling more sophisticated, automated approaches that yield high-quality and perceptually pleasing results.

This project report presents a comparative analysis of three advanced image colorization techniques: Convolutional Neural Networks (CNNs), Generative Adversarial Networks (GANs), and Diffusion Models. Each of these methods employs unique architectures and learning paradigms to address the challenges of image colorization. To objectively evaluate the performance of each model, we adopted three metrics: Learned Perceptual Image Patch Similarity (LPIPS), Peak Signal-to-Noise Ratio (PSNR), and Structural Similarity Index Measure (SSIM). These metrics provide a multi-faceted view of model performance.

Furthermore, this report reflects on a project proposal to specifically address the colorization of repetitive objects. This concept was explored to understand how well each model could handle images with recurring patterns and structures. However, this idea was eventually set aside in favor of a broader comparative analysis to provide a more generalizable understanding of each method's capabilities. By integrating comprehensive experimental evaluations with a theoretical study of each model, this report aims to shed light on the state-of-the-art in image colorization, and to deepen the understanding of image colorization methods.

2. Related Work

2.1. Convolutional Neural Network

In the realm of image colorization, Convolutional Neural Networks (CNNs) have marked a significant milestone, transforming the way we approach the task of infusing grayscale images with color. The architecture typically involves a series of convolutional layers that automatically and hierarchically extract and learn features from images at various levels of abstraction. For colorization, this means learning to associate specific patterns and textures with appropriate colors in a data-driven manner.

Zhang et al. utilized a CNN model [1], which autonomously colors grayscale images without additional user input. This model is particularly noteworthy for its use of a deep residual network trained to predict color channels, capitalizing on large-scale dataset training to capture rich color statistics of objects and scenes. Additionally, the network employs a pre-trained VGG network as a feature extractor to enhance its learning capabilities. The output part of the network utilizes multi-scale features and deconvolution layers to achieve refined colorization, rendering the generated color images visually more natural and realistic.

2.2. Generative Adversarial Network

Another common approach for colorization tasks is using generative models. In 2014, Goodfellow et al. [2] introduced a novel form of generative model known as Generative Adversarial Networks (GANs). The GANs are composed of two smaller networks, i.e., the generator and discriminator. The generator aims to generate outcomes that closely resemble the real data, while the discriminator's task is to differentiate between a sample sourced from the generator's model distribution and the original data distribution. Both these subnetworks are trained concurrently until the generator attains the ability to consistently produce outcomes that are undistinguishable by the discriminator from the original data distribution.

Nazeri et al. proposed a fully automatic colorization method based on GAN in [3], 2018. The architecture of their generator network is shown in Figure 1. They trained the proposed GAN on the CIFAR-10 and Places365 dataset. The generator model used a symmetric Unet-like architecture with contracting paths. For the discriminator, they followed a patch-gan architecture and used a sigmoid function to produce probability values to determine whether it was real or fake.

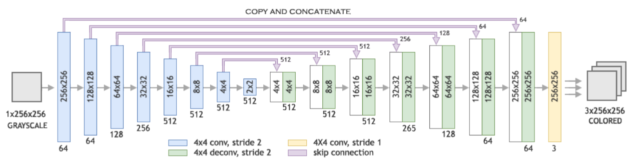


Figure 1. Unet architecture

2.3. Diffusion Model

In recent years, Diffusion Models have been garnering attention due to their impressive capabilities across a spectrum of generative applications. In [4] postulates that diffusion models are effectively generative models. These models typically involve a dual process: a forward process to integrate a data distribution, and a reverse procedure to glean learnings from it. Additionally, they deploy a sampling mechanism to fabricate novel data. The ingenuity of these models lies in their adaptability, showcasing excellent performance in various applications, such as image and text generation, voice synthesis, and in this case, image colorization.

Continuing our exploration of emerging techniques, [5]brings diffusion models and Unet together for the task of image colorization. This strategy centers on the generation of noise at random time intervals via the diffusion model before leveraging the exceptional capabilities of a specific network structure, namely Unet, to train and consequently derive a denoised, generated result. Drawing inspiration from this fusion, we have adapted and incorporated it into our current project.

2.4. Repetitive Objects

With our goal set on colorizing repetitive objects within an image, we took an interest in a method detailed in [6]. This method utilizes atoms, or indivisible pieces of objects, for detection and weeds out geometric inconsistencies. Guided by our tutor's advice, we strategized an approach to implement this method. However, we struggled to find an appropriate dataset and had to switch back to our original proposal of comparative analysis of different coloration models.

3. Methods

3.1. CNN

Our colorization approach employs a Convolutional Neural Network (CNN) that integrates a modified ResNet architecture with an upsampling mechanism. This design is specifically engineered to address the task of adding color to grayscale images by predicting the AB channels in the LAB color space, while keeping the L channel (lightness) derived directly from the input grayscale image.

3.1.1 ResNet for Feature Extraction

The foundational element of our ColorizationNet is a ResNet model. We adapt the ResNet architecture, originally comprising 18 layers, to accept a single-channel (grayscale) input rather than the standard three-channel RGB input. This adaptation involves modifying the first convolutional layer's weights by summing across the color channels and reshaping them to fit a single-channel input. This change allows the network to process grayscale images while leveraging the powerful feature-extraction capabilities of ResNet, trained on the ImageNet dataset with preexisting weights (ResNet18_Weights.IMAGENET1K_V1). The model utilizes the first six layers of the ResNet, which are specifically configured to extract mid-level features that are crucial for the subsequent colorization process.

3.1.2 Upsampling for Color Channel Prediction

Following feature extraction, the architecture transitions to a series of upsampling layers designed to progressively refine and enlarge the feature maps. This section of the network consists of alternating configurations of convolutional layers, batch normalization, and ReLU activation functions, which together enhance the stability and efficiency of the model during training.

3.1.3 Final Output

The final part of the upsampling section combines the upsampled features into two channels corresponding to the A and B color components of the LAB color space. The network outputs these two channels, which are then combined with the original L channel to reconstruct the full color image in LAB space. This output can be easily converted to the RGB color space for display or further processing.

3.2. GAN

The coloration of grayscale image can be regarded as an image-to-image translation task with one-channel input and three-channel output. Based on such background, we use RGB images directly and take the converted grayscale images as the input and output 3-channel RGB images.

3.2.1 Generator and Discriminator

We utilized the generator structure of [3] which is built based on TensorFlow. And we reimplement it with PyTorch. The input undergoes a gradual downsampling process through a sequence of contracting layers from the encoder. Subsequently, this process is reversed by a series of upsampling decoding layers to reconstruct the original input. This generator will predict the missing color channels from the grayscale image. The discriminator follows the PatchGAN design. The input image will be split into smaller patches when passing through the internal blocks. The final output layer produces a 2D map of predictions and takes an average as the discriminative prediction.

3.2.2 Loss Function

The loss function used in our GAN model contains three parts: adversarial loss, content loss, and perceptual loss. The adversarial loss forms the core of the GAN framework. This loss encourages the generator to enhance its outputs in the competition against the discriminator. To optimize the model's performance, we introduced two additional losses. For content loss we use smooth L1 loss and choose $\lambda_{content}$ as 100 to push the generator to follow the ground truth. Moreover, we set perceptual loss to compare the higher-level features, making the generated images not only pixel-wise accurate but also visually coherent in terms of style and context. We add $\lambda_{content} = 0.5$ to this loss term.

3.2.3 Training strategies

We set the learning rate as 0.0002 with a decay rate of beta1=0.5. The training process is designed to run for 20 epochs by default, allowing the model sufficient time to learn and adapt while avoiding overfitting. We monitored the overfitting with generated log files. To further improve the model's performance, we introduced the following techniques:

- Batch normalization: This technique normalizes the input of each layer to have a mean of zero and a variance of one. Doing so helps in stabilizing the learning process and significantly speeds up the convergence of the training.

- Leaky ReLU activation function: The Leaky ReLU allows a small, non-zero gradient (slope) when the unit is not active and the input is less than zero. This feature helps in maintaining the flow of gradients during the training process, preventing the problem of "dying neurons" and encouraging a more dynamic learning phase.

3.3. Diffusion Model

The aim of our approach is to generate a colorized version of a single-channel input. We've adapted the LAB color transformation from RGB format, which provides a more vast and effective color range suited for our tasks, thanks to its closer alignment with human vision perception and separation of lightness and color channels. The process sees a 3-channel LAB input—constructed from the original grayscale channel and noised AB channels—fed into the model, which then outputs a 2-channel prediction of the color channel.

Our neural network, the Gaussian Diffusion model, aids in this effort. The model's constructor initializes it with a denoise model and betas, computing the corresponding alpha values, cumulative products of alphas (alphasbar), and noise scale factors for each step of the diffusion process.

3.3.1 Noise Generation

The noise scale factors, a key component in the operation of the model, are critical in determining the intensity of the Gaussian noise added at each step of the diffusion process. The noise scales with the value of beta at each time step, allowing the model to inject different amounts of noise into the data at each step. This approach serves a dual purpose: larger amounts of noise quickly obscure data structures, promoting better generalization for unseen data, while smaller amounts of noise promote better fitting of the distributions but could undermine generalization. Empirically, we have found that scheduling smaller noise levels at earlier steps for exploration and higher levels at later steps for exploitation results in a better balance.

This model depends on our noise generation function to generate the Gaussian noise. The function fetches a tensor of normally-distributed random values matching the size of the image tensor x and residing on the same computation device. Then it rescales the noise according to the time step using the pre-computed noise scale factors, before returning the resulting noise tensor. The generated noise plays an integral role in training the denoising model to handle different patterns of colorization while maintaining a balance between data recreation and generalization requirements.

3.3.2 Denoising Model

Our image colorization project leverages the influential U-Net architecture, which gained initial popularity in biomedical image segmentation. This architecture stands out due to its symmetric expansive path, which increases precision in localization—a feature that's crucial for our colorization task.

Incorporated within our U-Net model are three principal sections coded in PyTorch - an encoder (downsampling path), a middle section, and a decoder (upsampling path). During the encoding stage, we sequentially apply convolution operations, LeakyReLU, Batch Normalization, MaxPooling, and Dropout to downsample the input image while concurrently elevating the distinction of feature representations. The middle section furthers this process via convolution operations, proliferating low-resolution encoder output into a more profound feature space. The decoder, in contrast, performs transpose convolutions to upscale the feature maps, a step allowing the model to assemble a higher-resolution, more detailed image as output.

4. Experiments

4.1. Dataset

We used the Animal FacesHQ (AFHQ) dataset to train our coloration models. The dataset contains photos of animal faces at 512 × 512 resolution. To restrict the training effort required to an acceptable level, we use only the cat domain and resize the images to 256 × 256.

Developing a fitting dataset for repetitive objects proved difficult for our project. Initial efforts using a diffusion model to generate images gave poor results, with blurry edges and messy, overlapped objects. We also tried to adapt the tutor's suggestion but it was hard to extract images with repeated objects from a large dataset. Further attempts to leverage Blender for creating simplistic repeated objects were unsuccessful due to our limited knowledge in texturing and rendering. These challenges led us to reconsider our approach. Instead of creating a specific repetitive object dataset, we utilized existing datasets and continue making progress within our capability.

4.2. Metrics Evaluation

To accurately assess the performance of our colorization models, we employ a combination of three widely recognized metrics: LPIPS, PSNR, and SSIM. Each of these metrics offers a unique perspective on image quality and similarity, enabling a comprehensive evaluation of the generated images against the original, full-color images.

4.2.1 Learned Perceptual Image Patch Similarity (LPIPS)

Learned Perceptual Image Patch Similarity (LPIPS) quantifies the perceptual differences between two images, utilizing deep learning models to approximate human visual sensitivity to various image distortions. This metric evaluates the similarity between the colorized image and its original counterpart by extracting and comparing deep features from pre-trained convolutional neural networks (CNNs). LPIPS has emerged as a preferred metric for tasks where the preservation of textural and color fidelity is paramount, as it aligns more closely with human perception than traditional metrics. The adoption of LPIPS in our evaluation strategy ensures that our model's performance is gauged not just on pixel-level accuracy but on perceptually relevant aspects, crucial for the subjective quality assessment of colorized images.

4.2.2 Peak Signal-to-Noise Ratio (PSNR)

Peak Signal-to-Noise Ratio (PSNR) is a traditional, widely utilized metric in the field of image processing for assessing the quality of reconstructed images. PSNR measures the ratio between the maximum possible power of a signal and the power of corrupting noise that affects its fidelity, expressed in decibels. A higher PSNR value indicates a higher quality of reconstruction, suggesting that the generated image is closer to the original image in terms of pixel-wise intensity. PSNR is particularly useful for evaluating the baseline accuracy of image colorization models, providing a clear, objective measure of the noise level and pixel accuracy across the colorized output.

4.2.3 Structural Similarity Index (SSIM)

The Structural Similarity Index (SSIM) is an advanced metric designed to improve upon traditional methods like PSNR by taking into account the perceived changes in structural information, luminance, and contrast of an image. Unlike PSNR, which evaluates images purely on pixel-level differences, SSIM assesses the visual impact of three characteristics of an image: luminance, contrast, and structure, thus providing a more holistic view of image quality. SSIM is particularly valuable for image colorization tasks as it accounts for the structural integrity and textural details of the images, ensuring that the colorization process preserves the essential attributes that contribute to the overall visual perception of the image. The inclusion of SSIM in our evaluation framework allows us to closely monitor the preservation of these critical image features, thereby ensuring the production of high-quality, visually pleasing colorized images.

4.2.4 Results

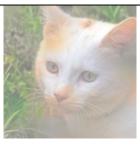
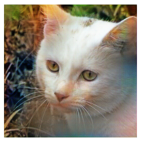
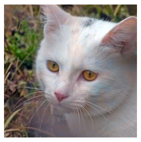
	generated image	real image
CNN		
GAN		
Diffusion Model		

Table 1. Visual Comparison of the Experiment

Method	LPIPS ↓	PSNR ↑	SSIM ↑
CNN	0.428	7.593	0.491
GAN	0.309	10.303	0.555
Diffusion Model	0.275	21.967	0.785

Table 2. Score Results of the Experiment

These results underscore the strengths and weaknesses of each model in the context of image colorization. The Diffusion model leads in overall image quality and similarity, making it an excellent choice for applications requiring high fidelity. The GAN provides a balanced approach with decent perceptual and structural outcomes, suitable for scenarios where a compromise between realism and computational efficiency is needed. Meanwhile, the CNN, despite its lower scores, remains a viable option for tasks where training data and model complexity are limiting factors.

5. Conclusions and Future Work

In our study, we conducted a rigorous comparison of several models for image colorization, including CNN, GAN, and diffusion model. Our findings revealed the strengths and weaknesses of these colorization methods, with each model offering unique advantages. The diffusion model, known for exploiting noise to enhance image quality, emerged as the superior model, delivering the highest fidelity in color and overall image quality. Meanwhile, GANs offered a respectable compromise between realistic colorization and computational efficiency. On the other hand, despite lower scores, the CNN proved viable for situations where there are limitations concerning training data and model complexity.

Reflecting on our experiment design, we acknowledge that it could have been more rigorous. A key aspect we noticed was the purity of our models, which appeared to be compromised as they incorporated elements from other methods. This mixing of different techniques within each model could potentially introduce variables that might have influenced our results. For a more accurate comparison in future studies, we plan to focus on testing each model in a more isolated and independent setup, thus preserving the integrity of each method's individual characteristics.

References

[1] R. Zhang, P. Isola, and A. A. Efros, “Colorful image colorization,” in *European Conference on Computer Vision (ECCV)*, 2016. Arxiv paper. Full text available at https://arxiv.org/abs/1603.08511. 1

[2] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, “Generative adversarial networks,” 2014. NeurIPS 2014 paper ID 5423. Full text available at https://arxiv.org/abs/1406.2661. 2

[3] K. Nazeri, E. Ng, T. Joseph, F. Z. Qureshi, and M. Ebrahimi, “Image colorization with generative adversarial networks,” 2018. Arxiv paper. Full text available at https://arxiv.org/abs/1803.05400. 2, 3

[4] Z. Chang, G. Koulieris, and H. Shum, “On the design fundamentals of diffusion models: A survey,” 2023. Arxiv paper. Full text available at https://arxiv.org/abs/2306.04542. 2

[5] A. Nir Zabari, Aharon Azulay, “Diffusing colors: Image colorization with text guided diffusion,” 2023. Arxiv paper. Full text available at https://arxiv.org/abs/2312.04145. 2

[6] B. Jonsson, S. Rivest, and P.-E. Forssén, “Automatic detection of repeated objects in images,” 2021. Link to full text: https://hal.science/hal-03126917/document. 2