

Image colorization: A survey and dataset

Authors	Anwar, Saeed;Tahir, Muhammad;Li, Chongyi;Mian, Ajmal;Khan, Fahad Shahbaz;Muzaffar, Abdul Wahab
Citation	S. Anwar, M. Tahir, C. Li, A. Mian, F. S. Khan, and A. W. Muzaffar, "Image colorization: A survey and dataset," Information Fusion, vol. 114, p. 102720, Feb. 2025, doi: 10.1016/J.INFFUS.2024.102720.
DOI	10.1016/j.inffus.2024.102720
Publisher	Elsevier
Download date	2026-06-07 05:34:14
Link to Item	https://hdl.handle.net/20.500.14634/258



Full length article

Image colorization: A survey and dataset

Saeed Anwar^{a,b,*}, Muhammad Tahir^c, Chongyi Li^d, Ajmal Mian^e, Fahad Shahbaz Khan^{f,g}, Abdul Wahab Muzaffar^h

^a Australian National University, Canberra, ACT, Australia

^b University of Canberra, Canberra, ACT, Australia

^c Department of Computer Science, National University of Sciences and Technology (NUST), Balochistan Campus, Quetta, Pakistan

^d Nankai University, China

^e The University of Western Australia, Australia

^f Mohamed Bin Zayed University of Artificial Intelligence, Abu Dhabi, United Arab Emirates

^g Computer Vision Laboratory, Linköping University, Sweden

^h College of Computing and Informatics, Saudi Electronic University, Saudi Arabia

ARTICLE INFO

Keywords:

Image colorization
Experimental survey
New colorization dataset
Colorization review
Deep learning
CNN model classification

ABSTRACT

Image colorization estimates RGB colors for grayscale images or video frames to improve their aesthetic and perceptual quality. Over the last decade, deep learning techniques for image colorization have significantly progressed, necessitating a systematic survey and benchmarking of these techniques. This article presents a comprehensive survey of recent state-of-the-art deep learning-based image colorization techniques, describing their fundamental block architectures, inputs, optimizers, loss functions, training protocols, training data, etc. It categorizes the existing colorization techniques into seven classes and discusses important factors governing their performance, such as benchmark datasets and evaluation metrics. We highlight the limitations of existing datasets and introduce a new dataset specific to colorization. We perform an extensive experimental evaluation of existing image colorization methods using both existing datasets and our proposed one. Finally, we discuss the limitations of existing methods and recommend possible solutions and future research directions for this rapidly evolving topic of deep image colorization. The dataset and codes for evaluation are publicly available at <https://github.com/saeed-anwar/ColorSurvey>.

1. Introduction

“Don't ask what love can make or do! Look at the colors of the world.” - Rumi

Colorization of images is a challenging problem due to the varying conditions of imaging that need to be dealt with via a single algorithm. The problem is also severely ill-posed as two out of the three image dimensions are missing; although the scene semantics may be helpful in many cases, for example, the grass is usually green, clouds are usually white, and the sky is blue. However, such semantic priors are uncommon for many manufactured and natural objects, e.g., shirts, cars, flowers, etc. Moreover, the colorization problem also inherits the typical challenges of image enhancement, such as changes in illumination, variations in viewpoints, and occlusions.

With the rapid development of deep learning techniques, various image colorization models have been introduced, and state-of-the-art

performance on current datasets has been reported. Diverse deep-learning models ranging from the early brute-force networks (e.g., [1]) to the recently carefully designed Generative Adversarial Networks (GAN) (e.g., [2]) have been successfully used to tackle the colorization problem. These colorization networks differ in many major aspects, including network architecture, network depth, loss functions, learning strategies, etc. This paper provides a comprehensive overview of single-image colorization and focuses on recent advances in deep neural networks for this task. To our knowledge, no survey of conventional or deep learning-based colorization techniques has been presented in the current literature. Our study concentrates on many important aspects, both systematically and comprehensively, in benchmarking recent advances in deep learning-based image colorization. **Contributions:** Our contributions are as follows

1. We thoroughly review image colorization techniques, including problem settings, performance metrics, and datasets.

* Corresponding author.

E-mail addresses: saeed.anwar@anu.edu.au (S. Anwar), m.tahir@nbc.nust.edu.pk (M. Tahir), lichongyi@nankai.edu.cn (C. Li), ajmal.mian@uwa.edu.au (A. Mian), fahad.khan@mbzuai.ac.ae (F.S. Khan), a.muzaffar@seu.edu.sa (A.W. Muzaffar).

<https://doi.org/10.1016/j.inffus.2024.102720>

Received 5 May 2024; Received in revised form 24 September 2024; Accepted 25 September 2024

Available online 30 September 2024

1566-2535/© 2024 The Authors. Published by Elsevier B.V. This is an open access article under the CC BY license (<http://creativecommons.org/licenses/by/4.0/>).

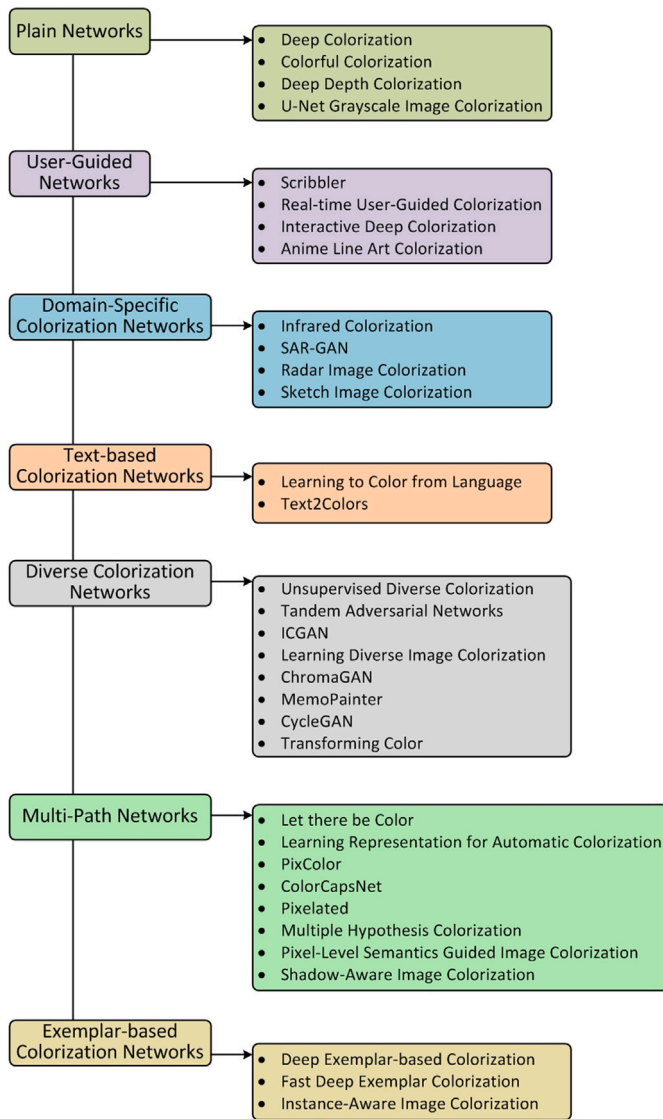


Fig. 1. Taxonomy: Classification of the colorization networks based on structure, input, domain, and type of network.

2. We propose a new taxonomy of colorization networks based on the differences in domain type, network structure, auxiliary input, and final output.
3. We introduce a new benchmark dataset, the Natural-Color Dataset (NCD), collected specifically for the colorization task.
4. We systematically evaluate state-of-the-art models on our Natural-Color Dataset.
5. We summarize the vital components of networks, provide new insights and discuss the challenges and possible future directions for image colorization.

Need for a Survey: Image colorization has been the focus of significant research efforts over the last two decades. Whereas most early image colorization methods were primarily influenced by conventional machine learning approaches [3–8] - the past few years have seen a massive shift in focus to deep learning-based models due to their success in several different application domains [1,9–16]. Automatic image colorization systems built upon deep learning have recently shown impressive performance. However, despite this success, no literature survey or review currently covers the latest developments in this field. Moreover, our goal is to streamline the direction of image

colorization research, put existing research in context, identify gaps, and point out the focus areas for the future. Therefore, inspired by surveys in deep image super-resolution [17], VQA [18], etc., we provide a comprehensive survey of deep image colorization.

Need for a Dataset: Datasets play a critical role in benchmarking the methods. The current datasets employed for evaluating algorithms in this field of research are not specific to colorization. The testing images may contain subjective colors such as walls, shirts, etc., and may have false colors; for example, the grass is blue. Similarly, the algorithms may be biased to training data, focus on a single object, color, or background, or spill the colors between objects or objects and the background. Hence, without a colorization-specific dataset, the evaluation may not accurately represent the algorithms. Therefore, we provide a colorization dataset and benchmark the state-of-the-art methods. We give further information about the dataset in the experimental section.

Problem Formulation: Colorization aims to estimate the RGB colors of a grayscale image, typically captured before color cameras were widely available and technological advancements were limited. Hence, this process is more of an image enhancement than an image restoration process. Another use for image colorization is to restore color images after they have been converted to grayscale or Y-Channel of YUV color space, for example, to save storage space or communication bandwidth. Therefore, in this case, a trivial formulation can be written as:

$$I_g = \Phi(I_{rgb}), \quad (1)$$

where $\Phi(\cdot)$ is a function that converts the RGB image I_{rgb} to a grayscale image I_g , for example as follows:

$$I_g = 0.2989 * I_r + 0.5870 * I_g + 0.1140 * I_b. \quad (2)$$

Typically, colorization methods aim to restore the color in YUV space, where the model needs to predict only two channels, i.e., U and V - instead of the three channels in RGB.

2. Single-image deep colorization

This section introduces various deep-learning techniques for image colorization. As shown in Fig. 1, these colorization networks have been classified into various categories regarding different factors, such as structural differences, input type, user assistance, etc. Some networks may belong to multiple categories; however, we assign each method to the most appropriate category. We also provide the strengths and weaknesses of each category in Table 1. Furthermore, we provide the timeline for all the articles, as shown in Fig. 2. Most of the work on image colorization was carried out with the advent of convolutional neural networks.

2.1. Plain networks

Similar to other CNN tasks, early colorization architectures were plain networks with a simple, straightforward architecture that stacked convolutional layers with no or naive skip connections. Networks that fall into this category are shown in Fig. 3 and discussed below. These networks are straightforward to implement; however, the learning process is slow and requires more training data.

2.1.1. Deep colorization

Deep colorization¹ [1] can be regarded as the first method to incorporate convolutional neural networks (CNNs) for the colorization of images. This method, however, does not fully exploit CNNs; instead, it includes joint bilateral filtering [53] as a post-processing step to remove artifacts introduced by the CNN network.

¹ Code available at <https://shorturl.at/cestD>

Table 1
The colorization algorithms summarization in terms of strength and weakness.

Category	Methods	References	Strengths	Weakness
Plain	<ul style="list-style-type: none"> • Deep Colorization • Colorful Colorization • Deep Depth Colorization • U-Net Grayscale Image Colorization 	<ul style="list-style-type: none"> • [1] • [19] • [20] • [21] 	<ul style="list-style-type: none"> • Simple architecture • Easy to implement 	<ul style="list-style-type: none"> • Limited performance • Gradient vanish problem • Create deeper networks challenging
User-guided	<ul style="list-style-type: none"> • Scribbler • Real-Time User-Guided • Interactive Deep Colorization • Anime Line Art Colorization 	<ul style="list-style-type: none"> • [22] • [23] • [24] • [25] 	<ul style="list-style-type: none"> • User choice incorporated • Diverse colorization possible 	<ul style="list-style-type: none"> • Colors are subject to user preference • Selected colors may not be natural • Requires human intervention
Domain-specific	<ul style="list-style-type: none"> • Infrared Colorization • SAR-GAN • Radar Image Colorization • Sketch Image Colorization 	<ul style="list-style-type: none"> • [26] • [27] • [28] • [29] 	<ul style="list-style-type: none"> • Domain knowledge incorporated • Faithful to the domain 	<ul style="list-style-type: none"> • Limited applications • Usually not Generalizable
Multi modal	<ul style="list-style-type: none"> • Learning to Color from Language • Text2Colors • Depth-Aware Colorization 	<ul style="list-style-type: none"> • [30] • [31] • [32] 	<ul style="list-style-type: none"> • Combines text/depth and visual information 	<ul style="list-style-type: none"> • Requires text input • Requires accurate text • Requires multi-modal data for training • Text may not be available always
Diverse colorization	<ul style="list-style-type: none"> • Unsupervised diverse colorization • Tandem adversarial networks • ICGAN • Learning diverse colorization • ChromaGAN • Memopainter • CycleGAN • Transforming Color • Image Colorization with Generative Color Prior 	<ul style="list-style-type: none"> • [33] • [34] • [35] • [36] • [37] • [2] • [38] • [39] • [40] 	<ul style="list-style-type: none"> • Produce different colorized images • Can be trained unsupervised 	<ul style="list-style-type: none"> • Difficult to obtain original colors • Dependent on the training data • Fails to distinguish similar local texture regions • Challenging to quantify the results
Multi-path	<ul style="list-style-type: none"> • Let there be color • Automatic colorization • PixColor • ColorCapsNet • Pixelated • Multiple Hypothesis Colorization • Pixel-Level Semantics Guided • Shadow-Aware Image Colorization 	<ul style="list-style-type: none"> • [41] • [42] • [43] • [44] • [45] • [46] • [47] • [48] 	<ul style="list-style-type: none"> • Solves vanishing gradient problem • Employed skip connections • Easy flow of information 	<ul style="list-style-type: none"> • May have complex architecture • Computationally expensive
Exemplar-based	<ul style="list-style-type: none"> • Deep Exemplar Colorization • Instance-Aware Colorization • Fast Deep Exemplar Colorization • CT Image Colorization 	<ul style="list-style-type: none"> • [49] • [50] • [51] • [52] 	<ul style="list-style-type: none"> • Guided via Exemplar images • Abundant examples available on internet 	<ul style="list-style-type: none"> • Results dependent on exemplar images • Examples may lack similar colors • Difficult deal with unusual colors • Challenging artistic colors

In the training stage, five fully connected linear layers are followed by non-linear activations (ReLU), and the loss function is the least-squares error. In their model, the number of neurons in the first layer depends on the dimensions of the feature descriptor extracted from the grayscale patch, while the output layer has only two neurons, i.e., the U and V channel corresponding color pixel values. During testing/inference, features are extracted from the input grayscale image at three levels, i.e., low-level, mid-level, and high-level. The features at the low level are the sequential gray values, at the mid-level are DAISY features [54], and at the high level, semantic labeling is performed. The patch and DAISY features are concatenated and then passed through the network. As a final step for removing artifacts, joint bilateral filtering [53] is performed.

The network takes as input a 256×256 grayscale image and is composed of five layers: one input layer, three hidden layers, and one output layer. The model is trained on 2688 images from the Sun dataset [55]. The images are segmented into 47 object categories, including cars, buildings, sea, etc., for 47 high-level semantics. Furthermore, 32-dimensional mid-level DAISY features [54] and 49-dimensional low-level features are utilized for colorization.

2.1.2. Colorful colorization

Colorful image Colorization CNN² [19] was one of the first attempts to colorize grayscale images. The network takes as input a grayscale

image and predicts 313 “ab” pairs of the gamut showing the empirical probability distribution, which is then transformed to “a” and “b” channels of the “Lab” color space. The network linearly stacks convolutional layers, forming eight blocks. Each block comprises two or three convolutional layers followed by a ReLU layer and Batch Normalization (BatchNorm [56]) layer. Instead of pooling, striding is used to decrease the size of the image. The input to the network is 256×256 , while the output is 224×224 ; however, it is resized later to the original image size.

Colorful image colorization removes the influence (due to the background, e.g., sky, clouds, walls, etc.) of the low “ab” values by re-weighting the loss based on the rarity of pixels in training. The authors term this technique class rebalancing. The framework utilized to build the network is Caffe [57], and ADAM [58] is used as the solver. The network is trained for 450k iterations with a learning rate of 3×10^{-5} , which is reduced to 10^{-5} at 200k and 3×10^{-6} at 375k iterations. The kernel size is 3×3 , and the feature channels vary from 64 to 512.

2.1.3. Deep depth colorization

Deep depth colorization³ (DE)²CO [20] employs a pretrained ImageNet [59] architecture for colorizing depth images. The system is primarily designed for object recognition by learning the mapping from the depths to RGB channels. The pretrained network weights are kept frozen, and only the last fully connected layer of (DE)²CO is trained to

² Code at <https://github.com/richzhang/colorization>

³ Available at <https://github.com/engharat/SBADAGAN>

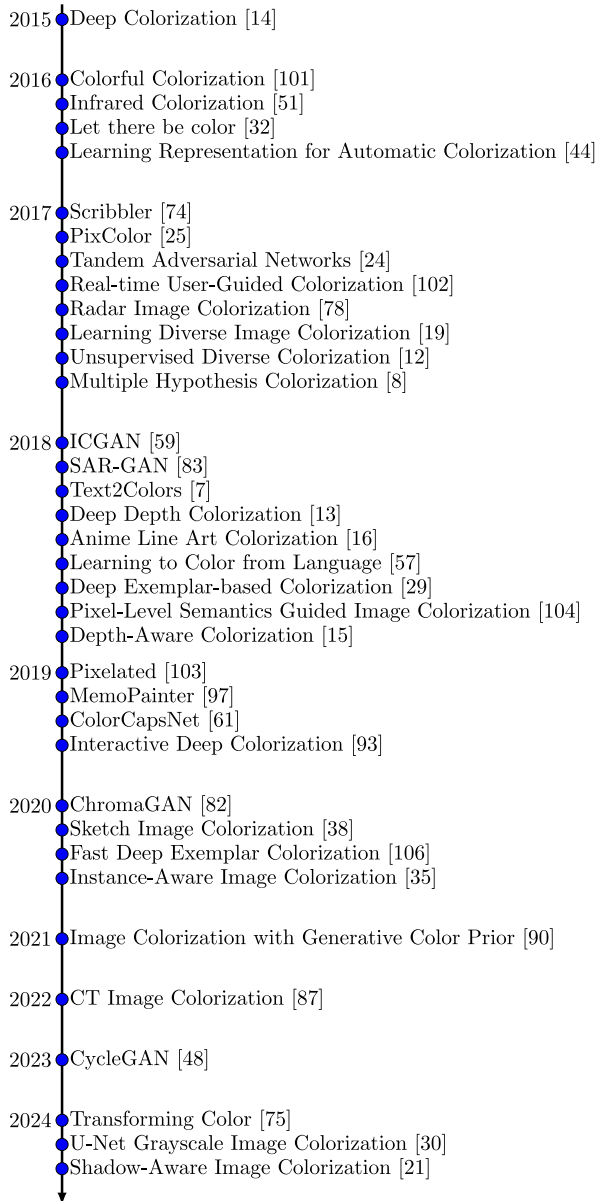


Fig. 2. The timeline chart indicates the year in which each method is made available.

classify the objects with a softmax classifier. The pretrained networks are merely used as feature extractors.

The input to the network is a 228×228 depth map reduced via convolution followed by pooling to $64 \times 57 \times 57$. Subsequently, the features are passed through a series of residual blocks, composed of two convolutional layers, each followed by batch normalization [56] and a non-linear activation function, i.e., leaky-ReLU [60]. After the last residual block, the output is passed through a final convolutional layer to produce three channels, i.e., an RGB image as an output. To obtain the original resolution, the output is deconvolved as a final step. When an unseen dataset is encountered, only the last convolutional layer is retrained while keeping the weights across all other layers frozen. (DE)²CO outperforms CaffeNet (a variant of Alexnet [61]), VGG16 [62], GoogleNet [63], and ResNet50 [64] under the same settings on three benchmark datasets, including Washington-RGBD [65], JHUIT50 [66], and BigBIRD [67].

2.1.4. U-Net grayscale image colorization

Hu et al. [21] utilized a fully convolutional neural network technique for grayscale image colorization tasks that involve the transformation of the color space followed by preprocessing of the output grayscale image and realization of an enhanced U-Net. The color space of input images is converted from RGB to Lab color space. Next, the output grayscale image is preprocessed using Heckbert's median cut method, which is used for gray-level quantization. Finally, the U-Net architecture is adopted to colorize the images. The U-Net network is trained on images from the Lab color space. Once trained, the network can generate pragmatic colors for grayscale images using the L component of the Lab color model after quantization with the median cut method. The outputs of each U-Net convolutional layer are obtained using the logistic activation function. The proposed method's performance was evaluated on the ImageNet database, and it achieved 84.81% accuracy.

2.2. User-guided networks

User-guided networks require input from the user either in the form of points, strikes, or scribbles, as presented in Fig. 4. Users can provide their input in real-time or offline. The following are examples of user-guided networks. User-guided colorization improves accuracy and provides control to users for direct input, enhancing complex and ambiguous regions colorization. Moreover, it may be useful for artistic purposes, precise historical restoration, handling small details and rare objects more effectively. However, all this is only possible due to increased user effort and a certain level of knowledge. This can be time-consuming and impractical for those without specialized knowledge. Although user-guided colorization is effective for individual images but these are less scalable for large datasets and can lead to inconsistent outputs when different users provide varying inputs for similar photos.

2.2.1. Scribbler

Sangkloy et al. [22] used an end-to-end feed-forward deep generative adversarial architecture⁴ to colorize images. To guide structural information and color patterns, user input in the form of sketches and color strokes is employed. The adversarial loss function enables the network to colorize the images more realistically.

The generator part of the proposed network adopts an encoder-decoder structure with residual blocks. Following the architecture of Sketch Inversion [68], the augmented architecture consists of three downsampling layers and seven residual blocks preceded by three upsampling layers. The downsampling layers apply convolutions of stride two, whereas the upsampling layers utilize bilinear upsampling to substitute the deconvolutional layers, contrary to Sketch Inversion. All layers are followed by batch normalization [56] and the ReLU activation function except the last layer, where the *TanH* function is used. A fully convolutional network with five convolutional layers and two residual blocks makes up the discriminator part of the proposed network. Leaky-ReLU [60] is used after each convolutional layer except the last one, where Sigmoid is applied. The model's performance is assessed qualitatively on datasets from three domains, including faces, cars, and bedrooms.

2.2.2. Real-time user-guided colorization

Zhang et al. [23] developed user interaction based on two variants, namely local hint and global hint networks, both of which utilize a common main branch for image colorization.⁵ The local hint network is responsible for processing the user input and yielding a color distribution, whereas the global hint network accepts global statistics in

⁴ <https://github.com/Pingxia/ConvolutionalSketchInversion>

⁵ <https://github.com/junyanz/interactive-deep-colorization>

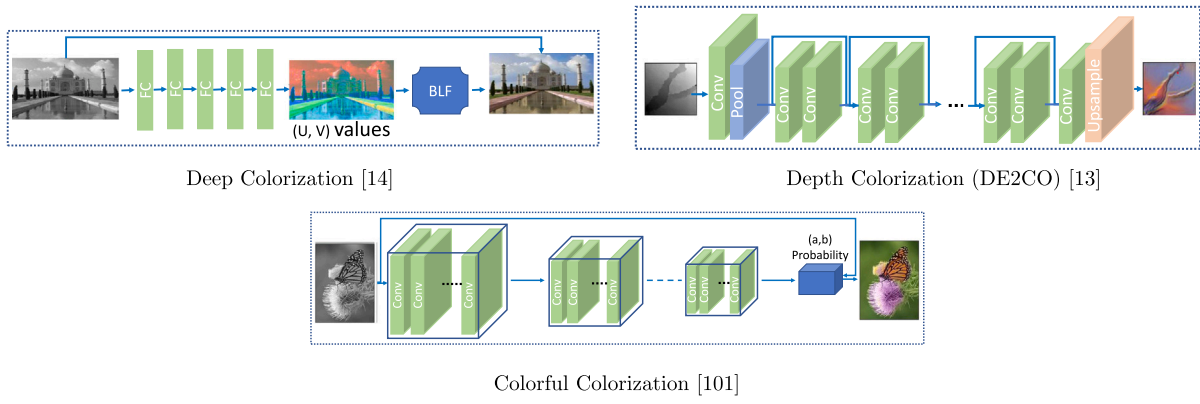


Fig. 3. Plain networks are the earlier models with convolutional layer stacking with no skip or naive skip connections.

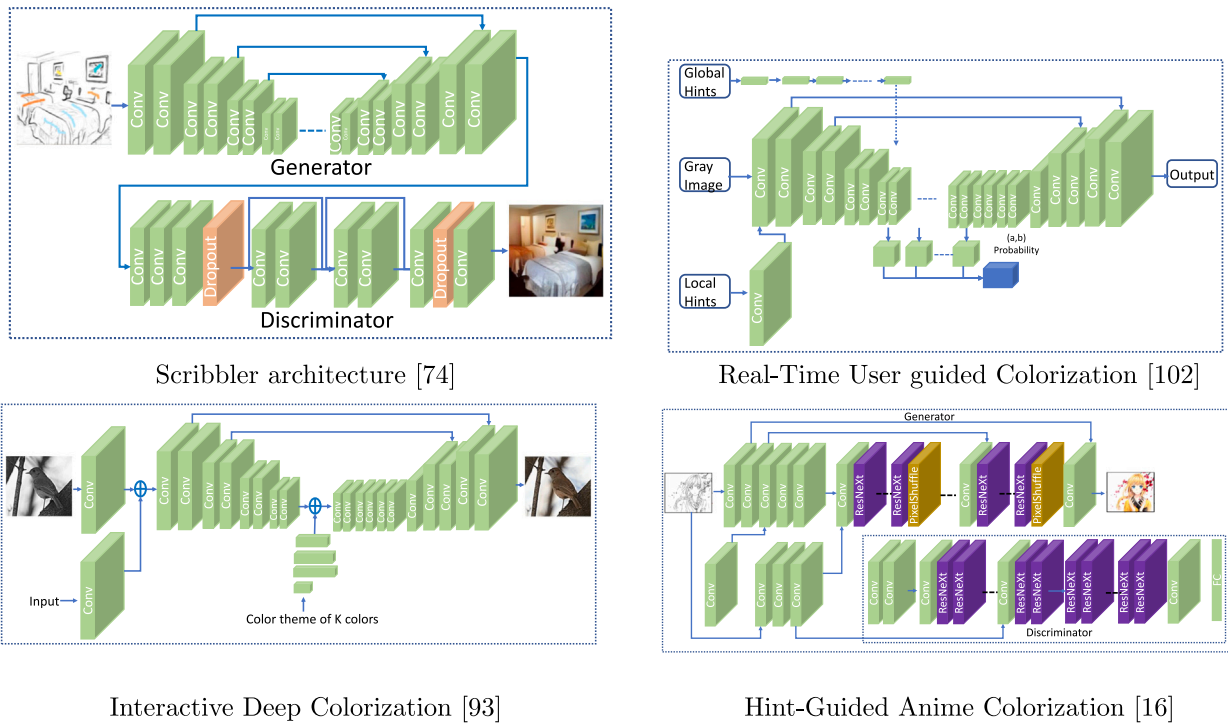


Fig. 4. User-guided networks are the ones that require a user to input the color at some stage of the network during colorization.

the form of a global histogram and average image saturation. The loss functions include Huber and regression.

The main network comprises ten blocks, made up of two or three convolutional layers, followed by the ReLU. Similarly, each block is succeeded by batch normalization as well. Feature tensors are continuously halved in the first four blocks while doubling the feature dimensions. The process is repeated in reverse order for the last four convolutional blocks. Moreover, a dilated convolution (with a factor of two) is applied for the fifth and sixth blocks. Symmetric shortcut connections are also established between different blocks to recover spatial information. The kernel size for all convolutions is set to 3×3 . However, at the last layer, a 1×1 kernel is used, mapping block ten and the final output.

2.2.3. Interactive deep colorization

To colorize grayscale images, Xiao et al. [24] developed an interactive colorization model based on the U-Net [69] architecture, which can simultaneously utilize global and local inputs. The network includes a

feature extraction module, a dilated module, a global input module, and a reconstruction module.

The feature extraction module (layers 2 to 14) receives the inputs, i.e., a grayscale image, local input, and gradient map merged via element-wise summation. Four convolution layers process the global input independently before combining it with the feature extraction module's output through element-wise summation. The dilated module takes the input from the extraction module (corresponding to convolutional layers ranging from 15 to 20). A reconstruction module, consisting of numerous deconvolution and convolution layers, further processes the output of the dilated module. The final step is combining the network's output with the input grayscale image to generate the colorized version. All the layers use ReLU except the final one, which employs a *TanH* activation.

The proposed model modifies the Huber loss to fulfill its requirements. The testing set is composed of randomly chosen 1k images from ImageNet [59]. The proposed model is trained for 300k iterations

utilizing the remaining pictures from ImageNet [59], combined with 150k images from Places dataset [70].

2.2.4. Anime line art colorization

Ci et al. [25] proposed an end-to-end interactive deep conditional GAN (CGAN) [71] for the colorization of synthetic anime line arts. The system operates on the user hints and the grayscale line art.

The discriminator is conditioned on the local features computed from a pre-trained network called Illustration2Vec [72]. The generator adopts the architecture of U-Net [69] that has two convolution blocks and the local feature network at the start. Afterward, four sub-networks with similar structures are employed, each of which is composed of convolutional layers at the front, followed by ResNeXt blocks [73] with dilation and sub-pixel convolutional (PixelShuffle) layers. LeakyReLU serves as the activation function for each convolutional layer, except the final one, which employs *TanH* activation. On the other hand, the discriminator is inspired by the architecture of SRGAN [74]; however, the basic blocks are replaced from the earlier mentioned generator to remove dilation, and an increased number of layers is used.

The generator employs the perceptual and adversarial losses, and the discriminator combines Wasserstein critic and penalty losses. The ADAM [58] optimizer is employed with $\beta_1 = 0.5$, $\beta_2 = 0.9$ and a batch size of four.

2.3. Domain-specific colorization

These networks aim to colorize images from different modalities, such as infrared, or domains, such as radar. Domain-specific colorization models perform better than general-purpose models because they understand the domain-specific subtleties, colors, and patterns. However, these networks have a limited capacity for generalization and frequently show inferior results on images outside of their trained domain. Furthermore, domain-specific colorization has limited usefulness, less applicability on more general images, and needs a significant amount of high-quality domain-specific data for training, which may only occasionally be available. We provide the details of such networks in the following sub-sections and show their architectures in Fig. 5.

2.3.1. Infrared colorization

An automatic Near Infrared (NIR)⁶ image colorization technique [26] was developed using a multi-branch deep CNN. NIR [26] is capable of learning luminance and chrominance channels. Initially, the input image is preprocessed and converted into a pyramid, where each pyramid level is normalized to zero mean and unit variance. Next, each structurally similar branch of NIR [26] is trained using a single input pyramid level without sharing weights between layers. All the branches are merged into a fully connected layer to produce the final output. Additionally, the mean input image is also fed directly to the fully connected layer.

The fundamental structure of NIR is inspired by [19]. Each branch is composed of stacked convolutional layers with pooling layers placed in the middle after regular intervals. The activation function after each convolutional layer is ReLU. To produce the colorized image, the raw output of the network is joint-bilaterally filtered, and the output is then further enhanced by incorporating high-frequency information directly from the original input image.

Each block has the same number of convolutional layers. The kernel size is 3×3 , while a downsampling of 2×2 is performed in the pooling layers. Similarly, the first block employs the same number of feature maps, i.e., 16, but increases the number by a factor of 2 after each pooling layer. Moreover, the authors developed a real-world dataset by capturing road scenes in summer with a multi-CCD NIR/RGB camera. The proposed model is trained on about 32 image pairs and tested on 800 images from the mentioned dataset.

2.3.2. SAR-GAN

To colorize Synthetic Aperture Radar (SAR) images, Wang et al. [27] proposed SAR-GAN. The network uses a cascaded generative adversarial network as its underlying architecture. The input SAR images are first denoised from speckles and then colorized to produce high-quality visible images. The generator of SAR-GAN [27] consists of a despeckling subnet and a colorization subnet. The despeckling subnet produces a noise-free SAR image, which the colorization subnet further processes to produce a colorized image.

The despeckling sub-network is made up of eight convolutional layers, an activation function, an element-wise division residual layer, and batch normalization [56]. First, the speckle component in the SAR image is estimated and forwarded to the residual layer. To generate a noise-free image, a residual layer equipped with skip connections performs component-wise division of the input SAR image by the estimated speckle. The colorization sub-network has a symmetric encoder-decoder architecture with three skip connections and eight convolutional layers. This lets the input and output of the network share low-level properties.

The ADAM [58] optimization technique is adopted for training the entire network. The discriminator component of SAR-GAN [27] uses a hybrid loss function developed by combining the pixel-level ℓ_1 loss with the adversarial loss. SAR-GAN [27] is tested on 85 out of 3292 synthetic images, as well as real SAR images.

2.3.3. Radar image colorization

Song et al. [28] developed a feature-extractor network and a feature-translator network for the colorization of single-polarization radar grayscale images.⁷ The first seven layers of the VGG16 [62] pre-trained on ImageNet [59] are used to construct the feature-extractor network. The output of the feature extractor network is a hyper-column descriptor obtained by concatenating the corresponding pixels from all layers with the input image.

The final hyper-column descriptor, thus obtained, is fed into the feature-translator network composed of five fully connected layers. As a final step, the feature-translator network uses the softmax function to obtain an output similar to that of classification, which is achieved by constructing nine groups of neurons representing nine polarimetrics. The ReLU activation function follows each convolutional layer in both networks. Furthermore, full polarimetric radar image patches are employed to train the feature extractor and feature translator.

2.3.4. Sketch image colorization

Lee et al. [29] proposed a reference-based sketch image colorization, where a sketch image is colorized according to an already-colored reference image. In contrast to grayscale images, which contain pixel intensity, sketch images are more information-scarce, which makes sketch image colorization more challenging.

In the training phase, the authors proposed an augmented self-reference generation method generated from the original image by color perturbation and geometric distortion. Specifically, an outline extractor first converts a color image into its sketch image. Then, the augmented self-reference image is generated by applying the thin plate splines transformation to the reference image. As the ground truth, the generated reference image contains most of the original image's content, thus providing full correspondence information for the sketch. Taking the sketch image and reference image as inputs, the network first encodes these two inputs by two independent encoders. Furthermore, the authors proposed a spatially corresponding feature transfer module to transfer the contextual representations obtained from the reference into the sketch's spatially corresponding positions (i.e., integrating the sketch's feature representations and its reference image). The integrated features are passed through residual blocks and

⁶ Code is available at <https://bit.ly/2YZQhQ4>

⁷ Code at <https://github.com/fudanxu/SAR-Colorization>

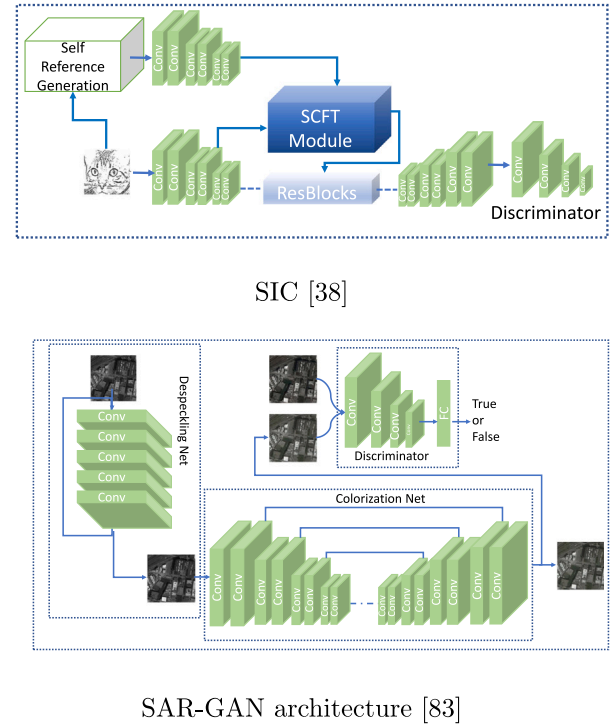
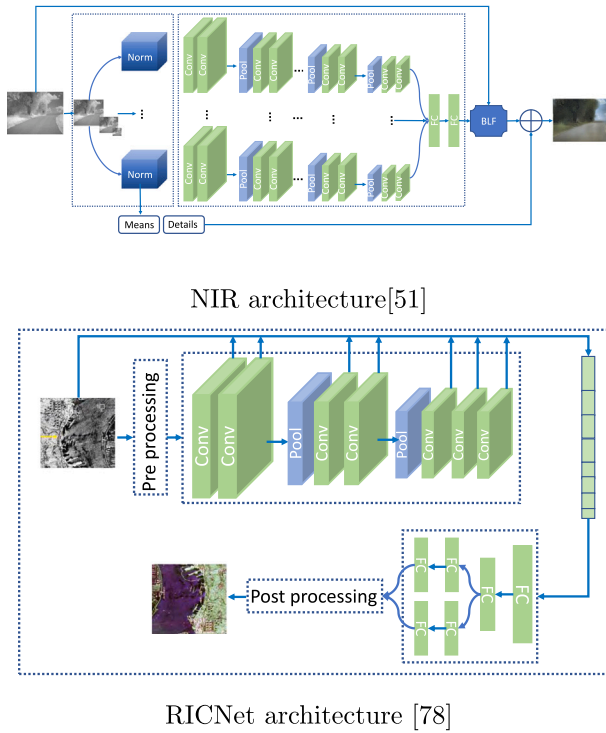


Fig. 5. Domain-specific colorization networks colorize images from different modalities such as infra-red, radar images.

a decoder to produce the colored output. During training, a similarity-based triplet loss, ℓ_1 reconstruction loss, adversarial loss, perceptual loss, and style loss are used to drive the learning of the proposed network.

The network has a U-net-like structure coupled with several residual blocks and a spatially corresponding feature transfer module between the encoder and decoder. The sketch and the color reference image with a size of 256×256 are fed to the network, which colorizes the sketch image.

2.4. Multi-modal colorization

These types of networks colorize the images based on an additional input, usually text or depth information. Users can specify distinct color schemes, thematic components, and in-depth color preferences. However, instructions written in complex terminologies may hinder the network performance, leading to inconsistent outcomes. Hence, the model may require more precise descriptions; otherwise, the output reliability may decrease. We classify the following models as multi-modal colorization networks. Fig. 6 presents the network architectures for this category.

2.4.1. Learning to color from language

To exploit additional text input for colorization, a language-conditioned colorization architecture⁸ [30] was proposed to colorize grayscale images with additional learning from image captions. The authors employed an existing language-agnostic architecture called FCNN [19], providing image captions as an additional input. FCNN [19] comprises eight blocks, each consisting of a sequence of convolutional layers followed by batch normalization. The authors experimented with the *CONCAT* network [75] to generate captions from images but settled on the *FILM* network [76] due to its small number of parameters. To obtain the final output, the language-conditioned weights of the

FILM [76] are utilized for affine transformation of the convolutional blocks' outputs.

Finally, the FCNN [19] and the two language-conditioned architectures, i.e., *CONCAT* and *FILM*, were trained on images from the MS-COCO dataset [77]. Automatic evaluations indicated that the *FILM* architecture achieves the highest accuracy. Moreover, crowd-sourced assessments validated the effectiveness of the models. The network's output is a 56×56 color image.

2.4.2. Text2Colors

The Text2Colors⁹ [31] model is comprised of two conditional generative adversarial networks: Text-to-Palette Generation Network (TPN) and Palette-based Colorization Network (PCN). TPN is responsible for constructing color palettes learned from the Palette-and-Text (PAT) dataset, which contains five color palettes for each of 10,183 textual phrases. PCN is responsible for colorizing the input grayscale image given the generated palette based on the input text.

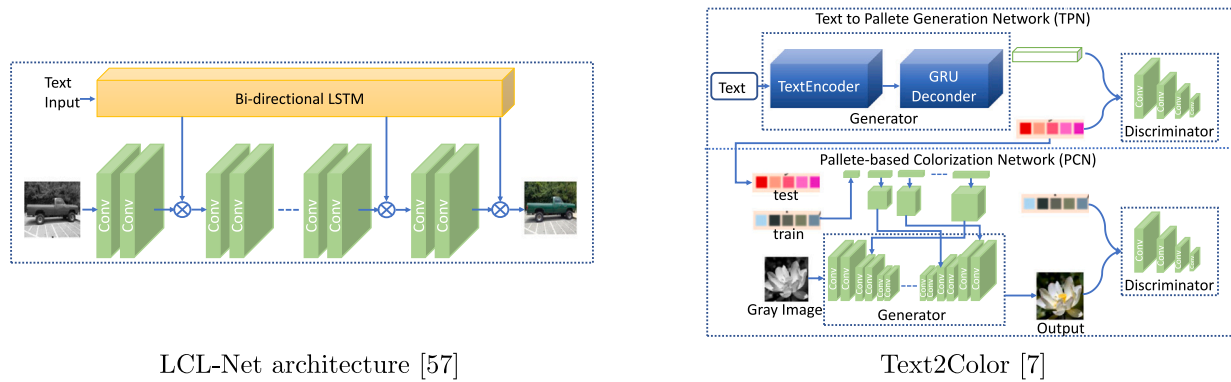
The TPN generator learns the pairing between text and color palette, while its discriminator simultaneously learns the color palette and text characteristics to identify the real palettes from the fake ones. The Huber is the loss function in the TPN network.

On the other hand, the generator of the PCN network has two subnetworks: the colorization network based on U-Net [69] to colorize the images and the conditioning network to apply the palette colors to the generated image. The PCN discriminator is based on the DCGAN architecture [78]. In the PCN's discriminator, first, the features from the input image and generated palette (by the TPN network) are jointly learned by a series of Conv-LeakyReLU layers. Then, a fully connected layer classifies the image as real or fake.

The discriminator and generator of TPN are first trained on the PAT dataset for 500 epochs and then trained on the ground truth image for 100 epochs. The trained generators of TPN and PCN are then used to colorize the input grayscale image at the testing stage using the input text's color palette. Adam optimizer is used with a learning rate of 0.0002 for all the networks.

⁸ Available at <https://github.com/superhans/colorfromlanguage>

⁹ Code is <https://github.com/awesome-davian/Text2Colors/>



LCL-Net architecture [57]

Text2Color [7]

Fig. 6. Multi-modal colorization networks are based on the text input or depth information with the grayscale image..

2.4.3. Depth-aware colorization

Chu & Hsu [32] proposed the fusion of color distributions estimated by an intensity-based prediction network and depth-based prediction network where the depth information contributes to the detection of boundaries between different objects. The proposed network consists of two parallel sequences of convolutional blocks where the output of each network is combined using a fusion function. The final color of each pixel is estimated using a mapping function. The intensity-based prediction network has eight convolutional blocks, with the first two blocks consisting of two convolutional layers while the subsequent five blocks consist of three convolutional layers. The eighth block consists of a deconvolutional layer and two convolutional layers. A ReLU activation function follows each layer. All the blocks have a batch normalization layer except the last block. Similarly, a depth-based prediction network has three convolutional blocks where the input to this network is a depth map of the input image. The first two blocks consist of two convolutional layers. ReLU follows each layer, while only the first block has a batch normalization layer. The third block consists of a single convolutional layer responsible for feature distribution into a specified number of bins.

2.5. Diverse colorization

Diverse colorization aims to generate different colorized images rather than restore the original color, as shown in Fig. 7. Diverse colorization is usually achieved via GANs or variational autoencoders. GANs attempt to generate the colors in a competitive manner, where the generator aims to fool the discriminator while the discriminator's goal is to differentiate between the ground truth and the generated colors. These models are useful for applications where diverse colors are possible and help explore more vibrant colors for a specific scene or item. However, they fail when the item's color is consistent. The output diversity may introduce ambiguity or inconsistency, and selecting the most appropriate colorization is challenging. Furthermore, the computational time and resource consumption also increase. We present the GANs employed for colorization in the following paragraphs.

2.5.1. Unsupervised diverse colorization

Cao et al. [33] proposed the use of conditional GANs for the diverse colorization of real-world objects.¹⁰ In the generator of the GAN network, the authors employed five fully convolutional layers with batch normalization and ReLU. Every generator layer concatenates the grayscale image to provide continuous conditional supervision for realistic results. Furthermore, noise channels are added to the first

three convolutional layers of the generator network to diversify the colorization outputs. Four convolutional layers and a fully connected layer make up the discriminator, distinguishing between the image's fake and real values.

During the convolution operations, the stride is set to 1 to keep the spatial information the same across all layers. The performance of the proposed method was assessed using the Turing test methodology. The authors provided questionnaire surveys to 80 subjects, asking them 20 questions regarding the results produced for the publicly available LSUN bedroom dataset [79]. The proposed model obtained a convincing rate of 62.6% compared to 70% for ground truth images. Additionally, a significance t-test generated a p -value of 0.1359, indicating that the generated colorized images are not significantly different from the real images.

For implementation, the authors opted for the TensorFlow framework. The authors selected a batch size of 64 and a learning rate of 0.0002 and 0.0001 for the discriminator and generator, respectively. The model was trained for 100 epochs using RMSProp as an optimizer. The network's output is 64×64 in size.

2.5.2. Tandem adversarial networks

For the colorization of raw line art, Frans [34] proposed two adversarial networks in tandem. The first network, namely the color-prediction network, predicts the color scheme from the outline, while the second network, called the shading network, produces the final image from the color scheme generated by the color-prediction network in conjunction with the outline. The color-prediction and shading networks have the same structure as U-Net [69] while the discriminator has the same structure as [33]. The adversarial training uses the discriminator formed by stacking the four convolutional layers and a fully connected layer at the end.

The convolutional layers are transposed once the density of the feature matrix reaches a certain level. The author also added skip connections between the corresponding layers to allow the gradient to flow through the network directly. Further, ℓ_2 and adversarial losses are incorporated in the color-prediction and shading-networks, respectively. The convolutional filter size is set to 5×5 with a stride of two for every convolutional layer in the network. Starting from an initial 64 feature maps, the number of feature maps is increased twice after every layer.

2.5.3. ICGAN

Image Colorization Using Generative Adversarial Networks abbreviated as ICGAN [35] proposed by Nazeri et al. has demonstrated better performance in image colorization tasks than traditional convolutional neural networks. Fully convolutional networks for semantic segmentation [80] inspire the baseline model and are constructed by replacing

¹⁰ Code is available at <https://github.com/ccyyatnet/COLORGAN>

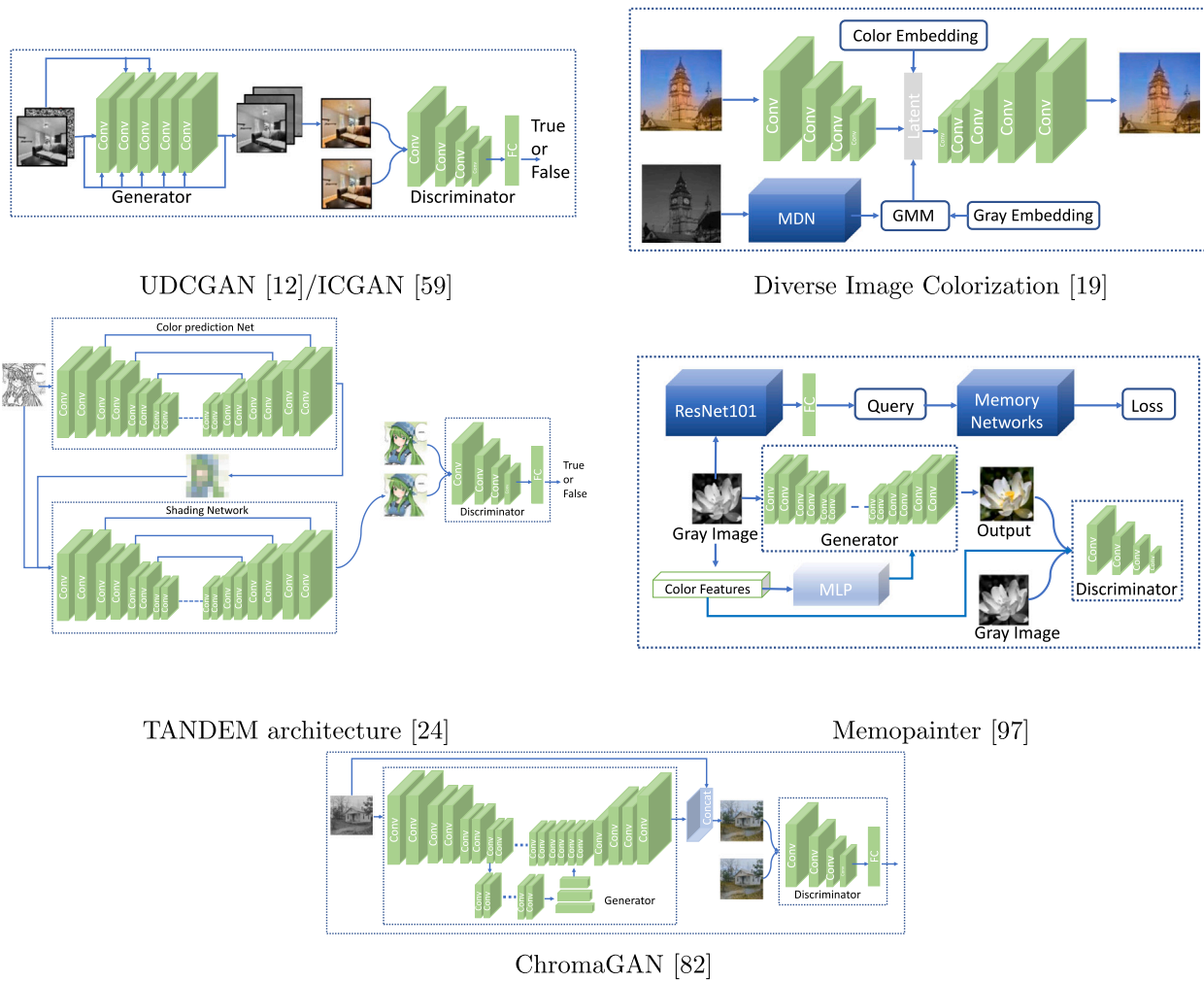


Fig. 7. Diverse colorization networks generate different colorized images instead of aiming to restore the original color only.

the fully connected layers of the network with fully convolutional layers. The basic idea is to downsample the input image gradually via multiple contractive encoding layers and then apply a reverse operation to reconstruct the output with many expansive decoding layers, similar to U-Net [69].

The ICGAN [35] generator takes grayscale input image instead of random noise, like traditional GANs. The authors also proposed the generator's modified cost function to maximize the discriminator's probability of being incorrect instead of minimizing the likelihood of being correct. Moreover, the baseline model's generator is used without any modifications, while the discriminator comprises a series of convolutional layers with batch normalization [56].

The filter size in each convolutional layer is 4×4 as opposed to the traditional 3×3 , and the activation function is Leaky-ReLU [60] with a slope value of 0.2. The final one-dimensional output is obtained by applying a convolution operation after the last layer, predicting the input's nature with a certain probability.

To train the network, the authors employed ADAM optimization [58] with weight initialization using the guidelines from [81]. The system's performance is assessed using CIFAR10 [82] and Places356 datasets [70]. Overall, the visual performance of the ICGAN [35] is favorable compared to traditional CNNs.

2.5.4. Learning diverse image colorization

Deshpande et al. [36] employed Variational Auto-Encoder (VAE) and Mixture Density Network (MDN) to yield multiple diverse yet

realistic colorizations for a single grayscale image.¹¹ The authors first employed VAE to learn a low-dimensional embedding for a color field of size $64 \times 64 \times 2$, and then used MDN to generate multiple colorizations.

The VAE architecture encoder consists of four convolutional layers with a kernel size of 5×5 and a stride of two. The feature channels start at 128 and double in the successive encoder layer. Batch normalization follows each convolutional layer, and ReLU serves as the activation function. The last layer of the encoder is fully connected. On the other hand, the decoder of the VAE architecture has five convolutional layers, each preceded by linear upsampling, batch normalization, and ReLU. The input to the decoder is a d -dimensional embedding, and the output is a $64 \times 64 \times 2$ color field. The convolutional kernel size is 4×4 in the first layer, and in the remaining, it is of size 5×5 . The activation function in all layers is ReLU, except in the last layer, where the activation function is $TanH$. The decoder halves the output channels at each layer.

MDN consists of twelve convolutional layers and two fully connected layers and is activated by the ReLU function, which is followed by batch normalization. During testing, embeddings sampled from the MDN output can be used by the decoder of the VAE to produce multiple colorizations. To enhance the overall performance of the proposed architecture, the authors designed three loss functions: specificity, colorfulness, and gradient.

¹¹ Code is available at <https://github.com/aditya12agd5/divcolor>

2.5.5. ChromaGAN

ChromaGAN¹² [37] exploits geometric, perceptual, and semantic features via an end-to-end self-supervised generative adversarial network. The proposed model can colorize the image realistically (actual colors) rather than merely focusing on aesthetic appeal by utilizing the semantic understanding of the real scenes.

The generator of ChromaGAN [37] is composed of two branches, which receive a grayscale image of size 224×224 as input. One of the branches yields chrominance information, whereas the other generates a class distribution vector. The network composition is as follows: the first stage is shared between branches that implement VGG16 [62] while removing the last three fully connected layers. The pre-trained VGG16 [62] weights are used to train this stage without freezing them. In the second stage, each branch follows its dedicated path. The first branch has two modules, each designed by combining a convolutional layer followed by batch normalization [56] and ReLU. The second path has four modules with the same composite structure (Conv-BN-ReLU) as the first branch but is further followed by three fully connected layers and provides the class distribution. The third stage fuses the outputs of these two distinct paths. The features then pass through six modules, each with a convolutional layer and ReLU with two upsampling layers.

The discriminator is based on PatchGAN [83] architecture, which holds high-frequency structural information about the generator's output by focusing on the local patches rather than the entire image. For five epochs, ChromaGAN [37] is trained on 1.3M images of ImageNet [59]. The optimizer used is ADAM, which has an initial learning rate of $2e^{-5}$.

2.5.6. MemoPainter: Coloring with limited data

MemoPainter¹³ [2] is capable of learning from limited data, which is made possible by the integration of external memory networks [84] with the colorization networks. This technique effectively avoids the dominant color effect and preserves the color identity of different objects.

Memory networks keep the history of rare examples, enabling them to perform well even with insufficient data. To train memory networks in an unsupervised manner, a novel threshold triplet loss was introduced by the authors [2]. In memory networks, the “key-memory” holds spatial information and computes cosine similarity against the input. Likewise, “value-memory” keeps a record of color information utilized by the colorization networks as a condition. Similarly, “age” keeps the time-stamp for each entry in the memory. The “key” and “value” memories are generated from the training data. The memory is updated by averaging the “key-memory” and a new query image owing to the distance between the color information of the new query image and the available top-1 “value-memory” element that lies below the threshold. Otherwise, a new record is stored for the input image color information.

The colorization networks are constructed using conditional GANs [71] consisting of a generator and discriminator, and color information is learned from RGB images during training. The generator inspired by U-Net [69] is composed of ten convolutional layers, while the discriminator is fully convolutional, consisting of four layers. The color feature is extracted and provided to the generator after being passed through the MLP.

During testing, the color information is retrieved from the memory networks and fed to the generator as a condition. The colorization networks employ adaptive instance normalization for enhanced colorization, considered a style transfer. The performance of MemoPainter [2] is evaluated on different datasets, including Oxford102 Flower [85], Monster [86], Yumi,¹⁴ Superheroes [2], and Pokemon.¹⁵

¹² Code available at <https://github.com/pvitoria/ChromaGAN>

¹³ <https://github.com/dongheehand/MemoPainter-PyTorch>

¹⁴ <https://comic.naver.com/webtoon/list.nhn?titleId=651673>

¹⁵ <https://www.kaggle.com/kvpratama/pokemon-images-dataset>

2.5.7. CycleGAN based image colorization

Li et al. [38] developed SS-CycleGAN, an automatic image colorization method that considers high-level and detailed semantics and spatial information of the objects in the image. SS-CycleGAN consists of two generators and two discriminators. One of the generators is used for translating grayscale images to color ones, and the other is used for translating color images to grayscale ones. Similarly, the two discriminators are separately used to discriminate the generated color and grayscale images and maintain the rationality of the produced color and grayscale images.

The authors implemented a patch discriminator for SS-CycleGAN built on a self-attention mechanism, which can administer the patch discriminator to concentrate on spatial structure information and the semantic relevance of colored objects in an image. The authors integrate the detail loss term with the joint loss function of SS-CycleGAN to ensure the details of the generated image remain consistent with the original image. Moreover, they further implemented a multi-scale cascaded dilated convolution module in the generators of SS-CycleGAN for the extraction of multi-scale features of local image patches to obtain spatial information on different colored objects.

2.5.8. Transforming color

Shafiq and Lee [39] proposed a novel technique for image colorization that integrates a VGG-based global feature extraction, a color encoder, a color transformer, and a GAN architecture. The transformer architecture extracts global information from the input image, and the GAN framework improves the image's overall subjective quality. After converting the RGB image into a Lab color space image, the input image is processed by a pre-trained VGG network, and the encoder captures high-level global semantic features. The output features of the VGG network are integrated with the encoder layers. Simultaneously, a color encoder employs convolutional layers to generate color information from a normal distribution. The fusion of color-encoded features at the bottleneck in the color transformer block and the global features in the encoder layers led to the integration of global and color-specific information. The Swin Transformer block captures the long-range dependencies and spatial relationships by processing the fused features. The proposed method can capture global features using two transformer blocks. Finally, the GAN's generator is composed of an encoder, a color transformer, a color decoder, and a discriminator. A combination of loss functions, including perceptual loss, adversarial loss, and color loss, achieves enhanced colorization during the training phase. The proposed architecture analyzes local image regions, enabling the proposed method to capture fine details of the entire image.

2.5.9. Image colorization with generative color prior

Wu et al. [40] adopted the concept of GANs to propose their colorization network that primarily consists of a GAN encoder, a pre-trained GAN, and a colorization network. The encoder receives the grayscale image as input and produces a latent code as output, which the pre-trained GAN receives as input and produces a color image of the original input grayscale image. Additionally, prior features are also obtained from the intermediate layers. The pre-trained GAN first spatially aligns these features with the input grayscale image based on the correspondence between the input image and the color image it produces. Finally, the colorization network receives a grayscale input image and generates the two color channels, chrominance and luminance, by modulating multi-scale spatial features with spatially adaptive denormalization layers. The final colorized image is obtained by combining the chrominance and luminance channels. Overall, the colorization network comprises two downsampling, two upsampling, and six residual blocks.

2.6. Multi-path colorization networks

The multi-path networks follow multiple different parts to learn features at various levels or paths. The multi-path models colorize

images by learning different visual features and contexts across several branches. This multi-path technique allows the accurate interpretation of a range of features. On the other hand, its disadvantage is that it often demands a significant amount of memory and computing resources to manage the multiple branches and feature concatenation; hence, it usually takes more time to train the models. The following are examples of multi-path networks, and Fig. 8 provides their architectures.

2.6.1. Let there be color

Iizuka et al. [41] designed a neural network¹⁶ to primarily colorize the grayscale images and provide scene classification as a secondary task. The proposed system has two branches for learning features at multiple scales. Both branches are further divided into four subnets, i.e., the low-level features subnet, mid-level features subnet, global features subnet, and colorization network. The low-level subnet decreases the resolution and extracts edges and corners. Mid-level features come in handy when learning the textures.

Similarly, the global subnet computes a 256D vector to represent the image. The mid-level and global features are combined and fed into the colorization network to predict the chrominance channel. The final output is restored to the original input resolution. The global features assist the classification network in classifying the image scene. Both the colorization network and scene classification network are jointly trained.

The low-level features subnet consists of six convolution layers; the mid-level features subnet consists of two convolutional layers; and the global features network consists of four convolutional layers and three fully connected layers. The colorization network comprises a fusion layer, four convolutional, and two upsampling layers. To feed input to the network, the image is resized to 256×256 , and then the output is upsampled to its original resolution. The kernel is of size 3×3 ; feature channels vary between 64 and 512, and striding is used to reduce the feature map size. Moreover, the learning rate is determined automatically using ADADELTA [87], the network is optimized using stochastic gradient descent (SGD) [88], and the system is designed using Torch [89].

2.6.2. Learning representations for automatic colorization

Learning Representations for Automatic Colorization¹⁷ [42] aims to learn a mapping function taking per-pixel features as hypercolumns of localized slices from existing CNN networks to predict hue and chroma distributions for each pixel. The authors use VGG16 [62] as a feature extractor, discarding the classification layer and using a grayscale image as input rather than RGB. For each pixel, a hypercolumn is extracted from all the VGG16 layers except the last classification layer, resulting in a 12k channel feature descriptor fed into a fully connected layer of 1k channels, which predicts the final hue and chroma outputs.

The KL divergence is employed as a loss to predict distributions over a set of color bins. The input size is 256×256 , and the framework used is Caffe [57], where the network is fine-tuned for ten epochs, each taking about 17 h. Other parameters and optimizations are similar to VGG16 [62].

2.6.3. PixColor

PixColor, proposed by [43], employs a conditional PixelCNN to produce a low-resolution color image from a given grayscale image. A refinement CNN is then trained to process the original grayscale and the low-resolution color image to produce a high-resolution colored image. The color of the previous pixels determines the colorization of an individual pixel.

PixelCNN is composed of two convolutional layers and three ResNet blocks [64]. The first convolutional layer uses kernels of size 7×7 ,

while the last employs 3×3 . Similarly, ResNet blocks contain 3, 4, and 23 convolutional layers, respectively. The PixelCNN colorization network is composed of three masked convolutional layers: one at the beginning and the second at the end of the network, whereas a gated convolutional block with ten layers is surrounded by the gated convolutional layers.

The PixelCNN model is trained by applying a maximum likelihood function with cross-entropy loss. Then, the refinement network, composed of 16 convolutional layers followed by two bilinear upsampling layers, each with two internal convolutional layers, is trained on the ground-truth chroma images downsampled to 28×28 . The network ends with three convolutional layers, where the final layer outputs the colorized image.

2.6.4. Colorize capsule network

Colorize Capsule Network (ColorCapsNet) [44] is built upon CapsNet [90] with three modifications. Firstly, the single convolutional layer is replaced by the first two convolutional layers of VGG19 [62] and initialized via its weights. Secondly, batch normalization [56] is inserted in between the first two convolutional layers. Thirdly, the capsule layer reduces the number of capsules from ten to six. The architecture of ColorCapsNet is similar to an autoencoder. The latent space processes the colorization-specific hidden variables between the encoder and decoder.

To train the model, the input RGB image is first converted into the CIE Lab colorspace, and extracted patches are fed to the network to learn the color distribution. The output is in the form of colorized patches, combined to obtain the complete image in Lab colorspace and later converted to RGB colorspace.

The difference between real and generated images is minimized using the mean squared error loss. ColorCapsNet is trained on ILSVRC 2012 to learn objects' general color distribution, which is then fine-tuned using the DIV2K dataset [91]. ColorCapsNet shows comparable performance to other models despite its shallow architecture. Adam is used as the optimizer with a learning rate of 0.001 and different kernel sizes.

2.6.5. Pixelated

Pixelated, introduced by Zhao et al. [45], is an image colorization model guided by pixelated semantics to keep the colorization consistent across multiple object categories. The network architecture is composed of a color embedding branch and a semantic segmentation branch. The network is built from gated residual blocks containing two convolutional layers, a skip connection, and a gating mechanism.

Three components make up the learning mechanism. *Firstly*, an autoregressive model is adopted to utilize pixelated semantics for image colorization. The model uses a shared CNN to generate per-pixel distributions through a conditional Pixel-CNN, achieving colorization diversity. *Secondly*, semantic segmentation is incorporated into color embedding by introducing atrous spatial pyramid pooling at the top layer capable of extracting multi-scale features using several parallel filters. The output is obtained by fusing multi-scale features. For semantic segmentation, the loss function is the cross-entropy loss with the softmax function. *Thirdly*, to produce better and more accurate color distributions, pixelated color embedding is concatenated with semantic segmentation to create the semantic generator.

Color embedding, semantics, and color generation losses are combined during training. The network performance is evaluated on the Pascal VOC2012 [92] and COCO-stuff [77] datasets. The images are rescaled to 128×128 for use in the network.

2.6.6. Multiple hypothesis colorization

Mohammad & Lorenzo [46] developed a multiple hypothesis colorization architecture, producing multiple color values for each pixel

¹⁶ https://github.com/satoshiizuka/siggraph2016_colorization

¹⁷ Code is available at <https://github.com/gustavla/autocolorize>

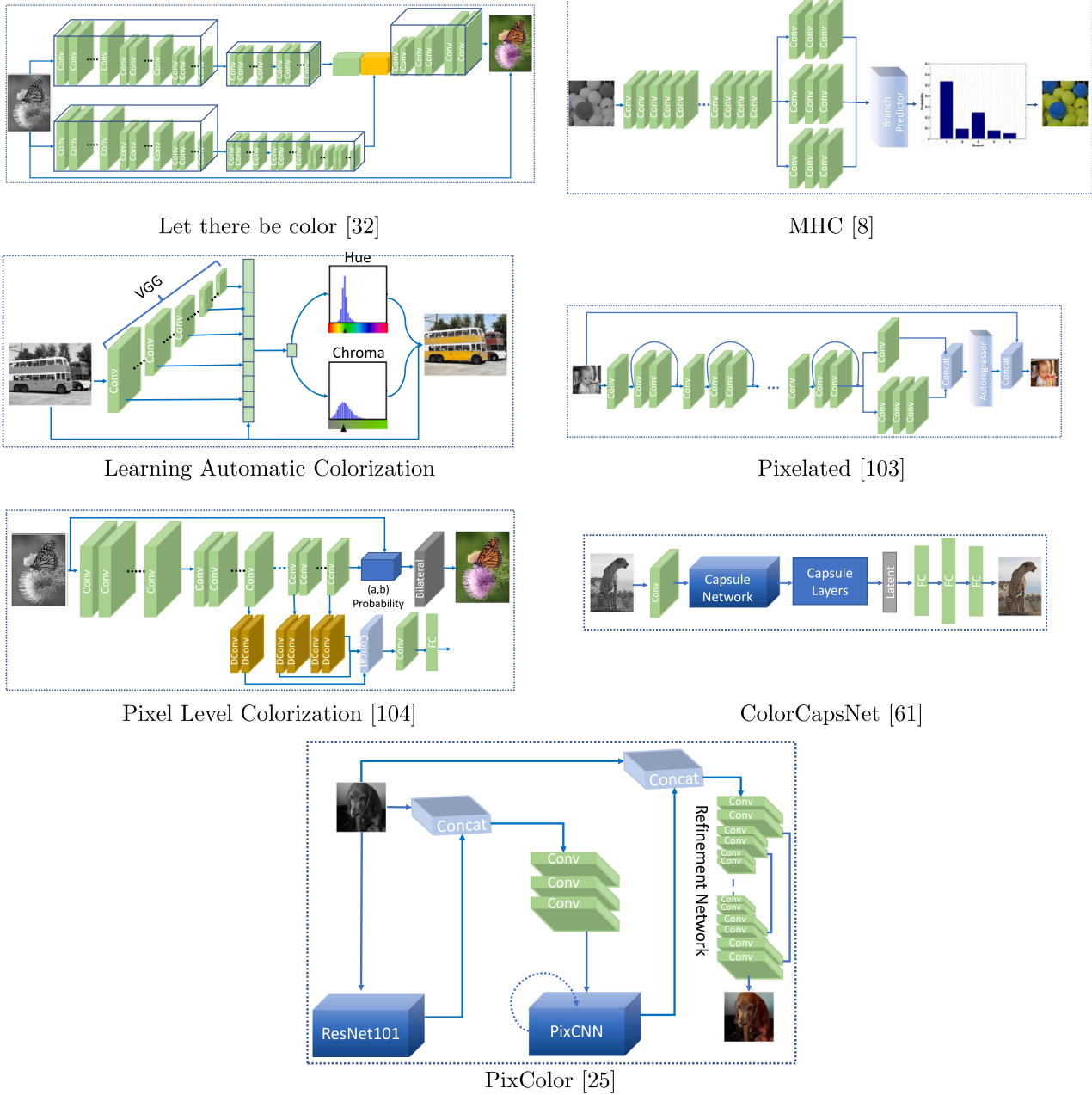


Fig. 8. Examples of multi-path networks. These colorization networks learn different features via several network paths.

of the grayscale image. The low-cost colorization is achieved by storing the best pixel-level hypothesis. A common trunk consisting of convolutional layers computes the shared features. The trunk is then split into multiple branches, where each branch predicts color for each pixel. The main trunk's layers and subsequent branches are fully convolutional.

The authors developed two architectures, one for CIFAR100 and the other for ImageNet [59]. The architecture for the CIFAR100 dataset is composed of 31 blocks preceded by a stand-alone convolutional layer. The number of channels in different layers across multiple blocks varies from 64 to 256. In contrast, the architecture for ImageNet [59] is composed of 21 blocks. The number of channels across different layers inside blocks varies from 32 to 1024. Although the two models have a different number of blocks, the block structure remains the same. Both networks also utilize residual connections. Moreover, batch normalization is performed in all blocks, and ReLU is applied to all layers except the last of each block.

Softmax was employed as a loss function. The CIFAR100 model was trained for 40k iterations with a learning rate of 0.001, dropping it after 10k and 20k iterations by a factor of 10. Similarly, the ImageNet [59] model was trained for 120k iterations with an initial learning rate of 0.0005 that is dropped after 40k and 80k iterations by a factor of 5.

2.6.7. Pixel-level semantics guided image colorization

A hierarchical network comprised of a semantic segmentation branch and a colorization branch was developed by Zhao et al. [47]. The two branches share the first four convolutional layers to learn low-level features. Four more convolutional layers extend the colorization branch, while three deconvolution layers come from the segmentation branch. The output of the deconvolutional layers is concatenated and passed through the final convolutional layer to produce class probabilities.

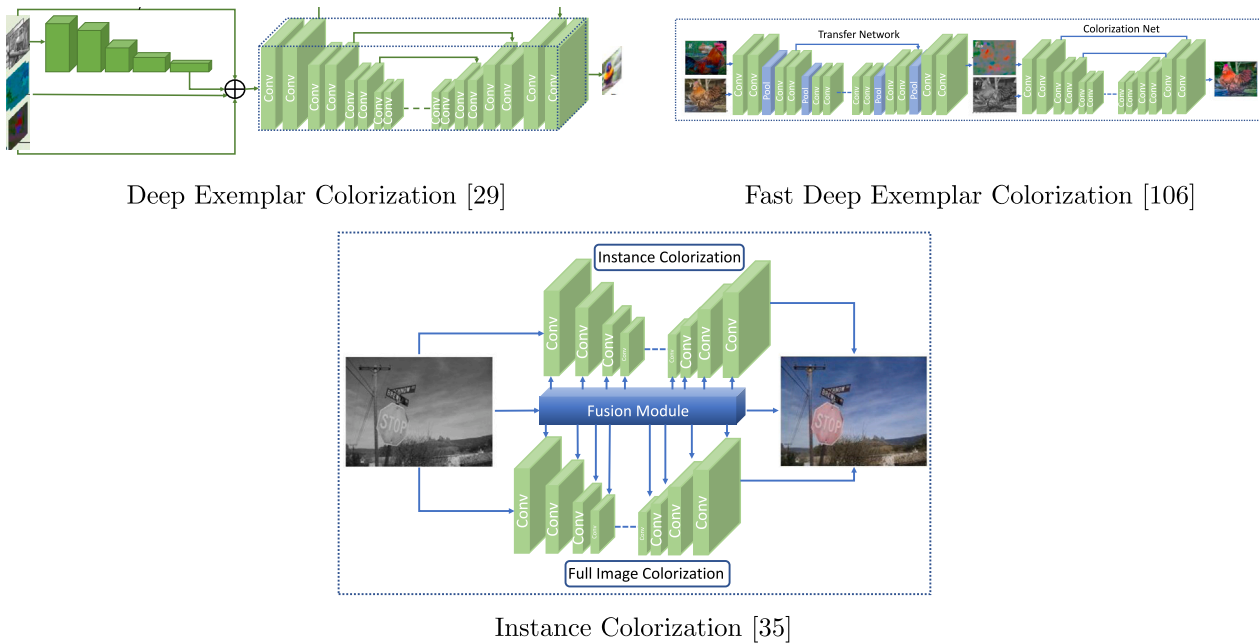


Fig. 9. Exemplar-based colorization networks. These networks imitate the colors of the input example image provided along with the grayscale image.

Semantic segmentation is achieved through weighted cross-entropy loss and softmax function, whereas colorization is performed via multinomial cross-entropy loss with class rebalance. During the training, the network jointly optimizes the two tasks. A joint bilateral upsampling layer is introduced during the testing to generate the final output.

The model was trained on 10,582 for semantic segmentation and tested on 1449 images from the PASCAL VOC2012 dataset for 40 epochs. Similarly, 9k images were used for training and 1k for testing on the COCO-stuff dataset [77] for 20 epochs.

2.6.8. Shadow-aware image colorization

Duan et al. [48] introduced a novel end-to-end dual-branch shadow-aware image colorization network that accurately colorizes shadow areas by incorporating shadow information with semantic understanding and preserves saturated colors. The shadow-aware block helps identify the shadow areas due to its ability to integrate shadow-specific information. The proposed methodology is capable of colorizing shadow and non-shadow areas separately.

The color information colorizes the non-shadow areas of the input image, whereas colorless information of shadow areas is used to colorize the shadow areas of the image. Separate processing of colorizing the shadow and non-shadow areas preserves the saturation and diversity of the non-shadow regions. Similarly, the shadow areas are colorized using colorless shadow information so that the overall saturation of the image is not hampered. The two-branch network has a shadow detection branch and a shadow-aware colorization branch. The shadow detection branch is an encoder that uses transformers to pull out multi-level features, and the shadow-aware colorization branch is another encoder that pulls out shadow-aware color features. Both the encoders share features that lead to effective colorization of the input image.

2.7. Exemplar-based colorization

Exemplar-based colorization utilizes the colors of example images provided along with input grayscale pictures and uses them to learn color distributions from related photos to improve the colorized output. When the exemplars closely match the content of the target pictures, they can capture subtle color changes and characteristics and generate compelling results. However, its effectiveness depends highly on the

exemplars. Therefore, an inadequate or mismatched exemplar might provide an inferior colorized image. It is also expensive to find and process the reference images. Furthermore, the technique may only be applicable if suitable exemplars are available. Fig. 9 presents the example networks for exemplar-based colorization. We provide more details about the networks in the subsequent subsections.

2.7.1. Deep exemplar-based colorization

Deep exemplar-based colorization¹⁸ [49] transfers the colors from a reference image to the grayscale one. The aim here is not to colorize images naturally but to provide diverse colors to the same image. The system consists of two subnetworks: the Similarity subnetwork and the Colorization subnetwork.

The similarity subnet takes the target and reference luminance channels aligned via Deep Image Analogy [93]. The authors use the standard features of VGG19 [62] after each block, resulting in five levels of coarse to fine feature maps. The features are upsampled to the same size. The similarity subnet computes a bidirectional similarity map using discrete cosine distance. Furthermore, the colorization subnet concatenates the grayscale image, chrominance channels, and the calculated similarity maps. U-Net [69] inspires the structure of the colorization subnet.

The loss function was ℓ_2 for a combination of chrominance channels and perceptual loss. For ten epochs, the network was trained using the ADAM [58] optimizer in the Caffe [57] framework, with a learning rate of 10^{-3} . This rate was lowered by 0.1 after 33% of training.

2.7.2. Fast deep exemplar colorization

Current exemplar-based colorization methods suffer from two challenges: (1) they are sensitive to selecting reference images, and (2) they have high time and resource consumption. Inspired by stylization characteristics in feature extraction and blending, Xu et al. [51] proposed a stylization-based architecture for fast deep exemplar colorization. The proposed architecture consists of a transfer sub-net that learns a coarse chrominance map (ab map in CIE Lab color space) and a colorization sub-net that refines the map to generate the final colored result. The proposed method seeks to produce realistic colorization

¹⁸ <https://github.com/msracver/Deep-Exemplar-based-Colorization>

results in real time, regardless of the semantic relationship between the input and exemplar image.

More specifically, an encoder–decoder structure is used for the transfer sub-net, which takes the target-reference image pairs as the input and output initial ab map. The pre-trained VGG19 module (from the $conv_1$ layer to the $conv_4$ layer) is treated as the encoder and a symmetrical decoder for image reconstruction. In addition, the fast Adaptive Instance Normalization (AdaIN) [94] is utilized after convolutional layers to accelerate feature matching and blending. The ab map generated by the transfer sub-net is inaccurate and has some artifacts. To refine it, a colorization sub-net that adopts an analogous U-Net structure is designed, taking a known luminance map and an initial chrominance ab map as input.

In the original implementations, the transfer sub-net is trained on the Microsoft COCO dataset [77] by minimizing the sum of the ℓ_2 loss, while the colorization sub-net is trained on the ImageNet dataset [59] by minimizing the Huber loss [95].

2.7.3. Instance-aware image colorization

The existing colorization models usually fail to colorize the images with multiple objects. To solve this issue, Su et al. [50] proposed an instance-aware image colorization method.¹⁹ The network includes three parts: (1) an off-the-shelf pre-trained model to detect object instances and produce cropped object images, (2) two backbone networks trained end-to-end, for instance, and full-image colorization, and (3) a fusion module to selectively blend features extracted from different layers of the two colorization networks. Specifically, a grayscale image is fed to the network as input. The network first detects the object bounding boxes using an off-the-shelf object detector. Then, each detected instance is cropped out and forwarded to the instance colorization network, for instance, colorization. At the same time, the input grayscale image is also sent to another instance colorization network (with the same structure but different weights) for full-image colorization. Finally, a fusion module fuses all the instance features with the full image feature at each layer until the output colored image is obtained. In the training phase, a sequential approach strategy is adopted that trains the full-image network followed by the instance network and finally trains the feature fusion module by freezing the above two networks. The smooth- ℓ_1 loss is used to train the network.

The authors use three datasets for training and evaluation, including ImageNet [59], COCO-Stuff [96], and Places205 [97]. Moreover, the images are resized to a size of 256×256 .

2.7.4. CT image colorization

Wang & Yan [52] presented an exemplar-based colorization scheme by following two distinct approaches. The first approach employs a ResNet model trained on diverse images for automatic colorization from grayscale CT images. In contrast, the second approach uses a VGG19 network trained on numerous reference images by transferring the style and color content to the target CT imagery. The ResNet-based network contains eight convolutional blocks, with each of blocks two to six consisting of a convolutional layer followed by a Leaky-ReLU activation, which ends in a batch normalization layer. Blocks one and seven consist of only a convolutional layer followed by a Leaky-ReLU activation function. Upsampling is performed after blocks five and six. The input to this network is a grayscale CT image, and the output is the two channels of the Lab color space, which are then combined with the lightness channel to produce the final colorized image. The VGG19 network learns reference images' style and color content for CT grayscale image colorization. The reference images include meat, rotten meat, lamb, beef, etc.

3. Experiments

3.1. Datasets

The data sets available for evaluation are the most commonly used in the literature for other tasks such as detection, classification, segmentation etc. The images are first converted to grayscale; then, colorization models are applied to analyze their performance. Hence, we provide a new dataset explicitly designed for the colorization task in the next section while we list the currently used datasets below.

- **COCO-stuff dataset** [96]: The Common Objects in COntext-stuff (COCO-stuff) dataset is constructed by annotating the original COCO dataset [77], which originally came with annotations of only things while neglecting the annotations of stuff. There are 164k images in the COCO-stuff dataset, which spans over 172 categories, including 80 things, 91 stuff, and one unlabeled class. The methods are evaluated using 5k images from the original validation set.
- **Places205** [98] has 20,500 test images from 205 categories. This dataset is commonly used to evaluate various colorization algorithms to investigate their relative performance. It only evaluates the transferability of algorithms and does not have a training partition.
- **PASCAL VOC dataset** [99]: PASCAL Visual Object Classes (PASCAL VOC) dataset has more than 11 000 images that are divided into 20 object categories.
- **CIFAR datasets** [82]: CIFAR-10 and CIFAR-100 are two subsets created and reliably labeled from 80 million tiny image dataset [100]. CIFAR-10 is comprised of 60k images equally distributed over ten mutually exclusive categories, with 6k in each image category. On the other hand, CIFAR-100 has the same images distributed over 100 categories, with 600 images assigned to each category. Each image in both the subsets is of size 32×32 pixels. In CIFAR-100, two-level labeling is used. There are 20 superclasses at the higher level, each further divided into five subclasses. Overall, 50k and 1k images comprise training and testing sets, respectively.
- **ImageNet ILSVRC2012** [59]: This dataset contains 1.2 million high-resolution training images spanning over 1k categories, where 50k images comprise the hold-out validation set. The split employed for testing, i.e., ctest10k, is from [42] containing 10k images for evaluation.
- **Palette-and-Text dataset** [31]: is constructed by modifying the data collected from color-hex.com, where users upload user-defined color palettes with label names of their choice. The authors first collected 47,665 palette-text pairs, removing non-alphanumeric and non-English words from the collection. After removing text-palette pairs that lack semantic relationships, the final curated dataset contains 10,183 textual phrases with their corresponding five-color palettes.

3.2. Evaluation metrics

The metrics typically used to assess colorization quality are either subjective or commonly used mathematical metrics, such as PSNR and SSIM [101]. Subjective evaluation is the gold standard for many applications in which humans determine the accuracy of the algorithm's output.

Although colorization uses subjective evaluation to some extent, it has two limitations: (1) scaling is extremely challenging, and (2) accurately determining the color is also very difficult. On the other hand, the mathematical metrics are general and may not provide the algorithms' accurate performance. We offer more insights into the metrics for colorization in the "Future Directions" section.

¹⁹ Code is available at <https://cg.v.cs.nthu.edu.tw/projects/instaColorization>

Table 2

Comparison of state-of-the-art methods for colorization in terms of PSNR, SSIM, and LPIPS on our ImageNet, COCOstuff, and Places205 datasets. Higher values of metrics indicate better performance.

Method	Imagenet ctest10k			COCOstuff			Places205		
	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑
Let there be Color [41]	0.200	23.64	0.917	0.185	23.86	0.922	0.146	25.58	0.950
Automatic Colorizer [42]	0.188	25.11	0.927	0.183	25.06	0.930	0.161	25.72	0.951
Colorful Colorization [19]	0.238	21.79	0.892	0.234	21.84	0.895	0.205	22.58	0.921
Real-Time Colorization [23]	0.145	26.17	0.932	0.138	26.82	0.937	0.149	25.82	0.948
Deoldify [102]	0.187	23.54	0.914	0.180	23.69	0.920	0.161	23.98	0.939
Fully-Automatic Colorization [103]	0.202	24.52	0.917	0.191	24.59	0.922	0.175	25.07	0.942
Instance-aware Colorization [50]	0.134	26.98	0.933	0.125	27.78	0.940	0.130	27.17	0.954

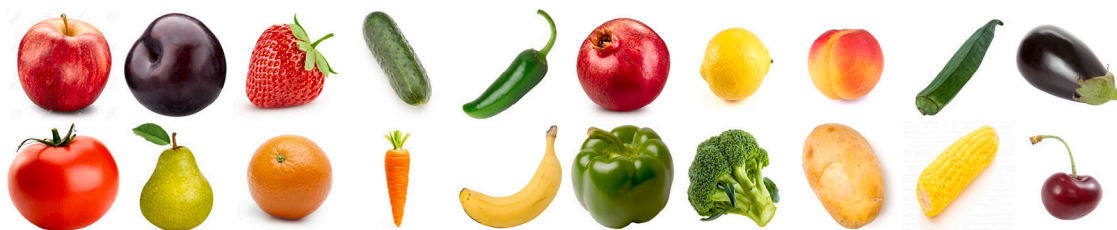


Fig. 10. Sample images for each category from our proposed Natural-Color dataset (NCD).

4. Comparisons

Qualitative Comparisons: We first compare the colorization methods on the existing datasets and then on our newly collected natural color dataset (NCD).

Existing Datasets Comparisons: The commonly used datasets for image colorizations are Places205 [98], ImageNet ILSVRC2012 [59] and COCO-stuff [96] datasets. The performances of various methods are given in Table 2, where the numbers generated by each method are more or less similar for SSIM and PSNR, but the results are better for instance-aware colorization in terms of LPIPS.

NCD Comparisons: The networks mentioned in Section 2 are evaluated on the peak signal-to-noise ratio (PSNR), the structural similarity index (SSIM) [101], patch-based contrast quality index (PCQI), and underwater image quality measure (UIQM) measures. Table 3 presents the results for each category for all measures on NCD. Real-time colorization [23] achieves the high performance of **21.93 dB** and **0.881** for PSNR and SSIM against other competitive measures. Furthermore, the PCQI performance of instance-aware colorization [50] and IQM achievement of colorful colorization [19] are higher than those of state-of-the-art methods. However, declaring one method against the other may not be a simple task due to the involvement of many various elements such as the number of parameters, depth of the network, the number of images for training, the datasets employed, the size of the training patch, the number of feature maps, the network complexity, etc. For a fair comparison, the only possible approach is to ensure that all methods have similar elements, as mentioned earlier.

Quantitative Comparisons: We present the visual colorization comparisons on fruits and vegetables in Figs. 11 and 12, respectively, for a few state-of-the-art algorithms. We can observe that most algorithms fail to recover the original natural colors for most images. Though Real-Time [23] and instance-aware [50] colorization algorithms provide consistent performance and colors closer to the original objects; however, the algorithms are still far from delivering accurate colorization performance. The experiments on the proposed Natural-Color Dataset (NCD) show the limitations of the state-of-the-art algorithms and encourage the authors to explore novel colorization techniques.

5. Limitations and future directions

Lack of Appropriate Evaluation Metrics: As mentioned earlier, colorization papers typically employ metrics from image restoration tasks,

which may not be appropriate for the task at hand. We also suggest comparing only the predicted color channels, i.e., U-channel and V-channel, instead of YUV. Similarly, it will be highly sought after to use metrics specifically designed to take color into account, such as PCQI [104] and UIQM [105].

PCQI stands for patch-based contrast quality index. It is a patch-based evaluation metric. It considers three statistics of a patch, i.e., mean intensity (p_m), signal strength or contrast change (p_c), and structure distortion (p_s) for comparison with ground truth. It can be expressed as

$$PCQI = p_m(x, y) \cdot p_c(x, y) \cdot p_s(x, y). \quad (3)$$

Similarly, UIQM is an abbreviation for the underwater image quality measure. It differs from earlier defined evaluation metrics because it does not require a reference image. Like PCQI, it also depends on three measures, i.e., image colorfulness, image sharpness, and image contrast. It can be formulated as:

$$IQM = w_1(ICM) + w_2(ISM) + w_3(ICoM), \quad (4)$$

where ICM, ISM, and ICoM stand for image colorfulness, image sharpness and image contrast, respectively, while “ w ” controls the weight of each quantity.

Lack of Benchmark Dataset: Due to the lack of purposefully built colorization datasets, researchers typically evaluate image colorization techniques on grayscale images from various datasets available in the literature. Originally, researchers collected the public datasets for tasks like detection, classification, etc., not for image colorization. The quality of the images may not be sufficient for image colorization. Moreover, the datasets contain objects, such as buses, shirts, doors, etc., that can take any color. Hence, such an evaluation environment is not appropriate for fair comparison in terms of PSNR or SSIM.

We aim to remove this unrealistic setting for image colorization by collecting images that are true to their colors. For example, a carrot will have an orange color in most images. Bananas will be either greenish or yellowish. Hence, our choice of images deliberately evaluates the method’s strength and any bias toward specific colors. Furthermore, we purposefully put a white background to test for any algorithm-induced colorization spill. We have collected 723 images from the internet distributed in 20 categories. Each image has an object and a white background. Our benchmark outlines a realistic evaluation scenario that differs sharply from those generally employed by image

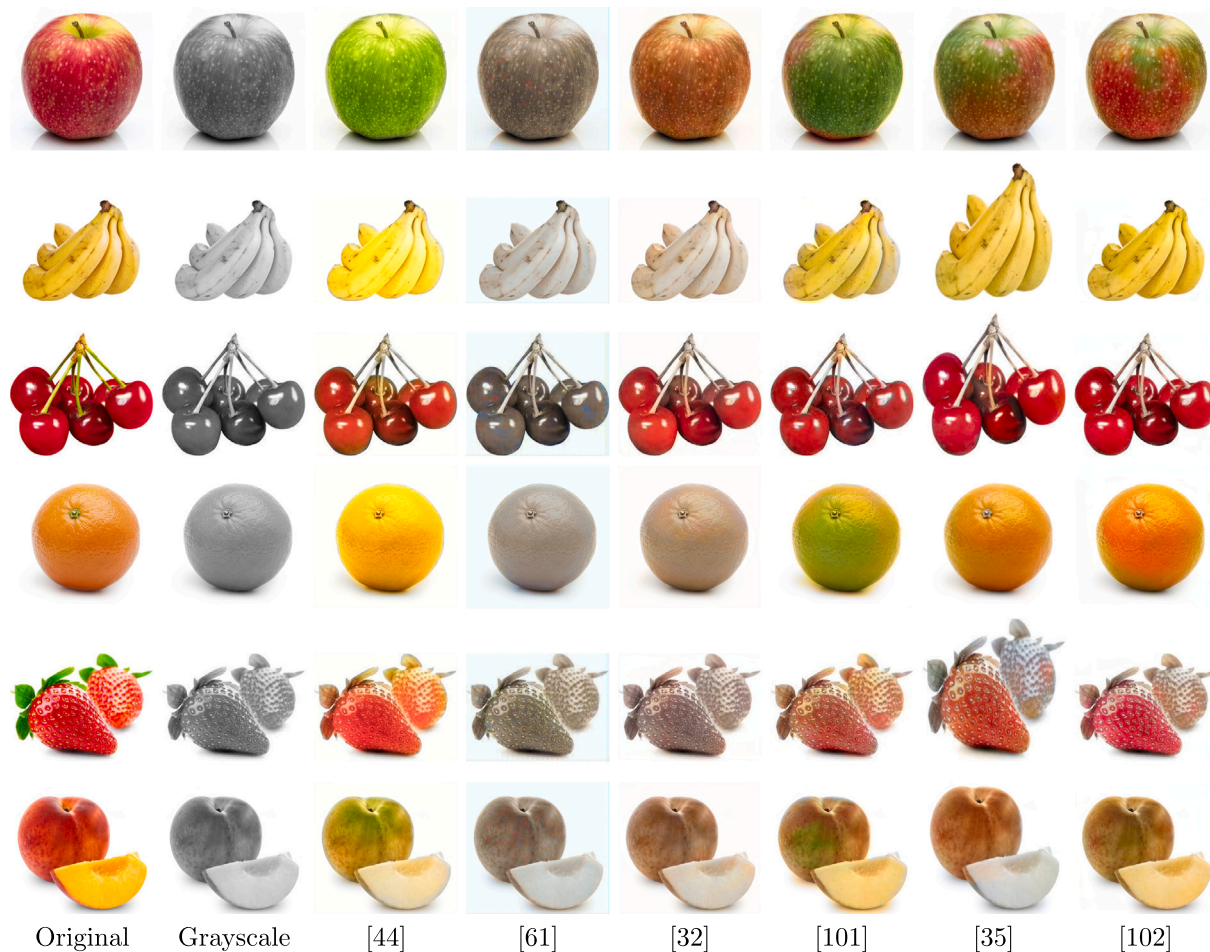


Fig. 11. Visual comparison of colorization algorithms on different fruit images from the Natural-Color Dataset. State-of-the-art colorization algorithms are unable to colorize the images effectively.

Table 3

Comparisons of the state-of-the-art methods for the colorization in terms of PSNR, SSIM, PCQI, and IQM on our Natural-Color dataset. The higher value of the metrics indicates better performance.

Category	No. of images	Automatic Colorizer [42]				ColorCapsNet [44]				Let there be Color [41]				Colorful Colorization [19]				Instance-Aware Colorization [50]				Real-Time Colorization [23]			
		PSNR	SSIM	PCQI	IQM	PSNR	SSIM	PCQI	IQM	PSNR	SSIM	PCQI	IQM	PSNR	SSIM	PCQI	IQM	PSNR	SSIM	PCQI	IQM	PSNR	SSIM	PCQI	IQM
Apple	39	16.66	0.772	0.881	1.009	17.64	0.625	0.825	0.869	20.17	0.820	0.918	0.944	18.71	0.792	0.922	1.095	21.94	0.903	0.918	0.918	21.27	0.888	0.915	1.104
Banana	44	17.83	0.792	0.937	0.688	16.43	0.598	0.820	0.652	16.74	0.733	0.936	0.634	22.22	0.894	0.948	0.813	22.14	0.889	0.947	0.643	23.34	0.923	0.948	0.780
Brinjal	35	27.21	0.849	0.946	1.179	26.32	0.720	0.876	1.015	27.74	0.866	0.939	1.094	24.52	0.829	0.946	1.304	25.59	0.843	0.949	1.134	27.56	0.855	0.944	1.242
Broccoli	35	18.57	0.870	0.895	1.419	18.48	0.717	0.865	1.232	18.59	0.816	0.918	1.301	19.34	0.842	0.920	1.483	20.11	0.851	0.964	1.431	19.54	0.854	0.918	1.435
Capsicum green	35	18.72	0.812	0.894	1.097	19.67	0.646	0.840	0.918	19.35	0.731	0.914	0.970	19.91	0.807	0.917	1.168	18.02	0.752	0.927	1.140	19.32	0.789	0.910	1.118
Carrot	39	23.10	0.929	0.931	0.921	17.65	0.655	0.852	0.748	19.40	0.826	0.927	0.825	21.16	0.917	0.939	1.026	22.00	0.908	0.932	0.832	22.62	0.937	0.936	1.005
Cherry	34	23.09	0.892	0.920	1.055	19.71	0.656	0.839	0.927	22.74	0.868	0.919	0.993	23.93	0.896	0.918	1.197	23.47	0.891	0.919	1.042	24.28	0.902	0.924	1.169
Chilli green	36	20.82	0.868	0.944	1.115	20.77	0.716	0.889	0.882	20.86	0.844	0.944	0.962	20.94	0.872	0.948	1.224	20.38	0.858	0.963	1.193	21.39	0.878	0.946	1.154
Corn	36	19.41	0.873	0.911	1.079	15.34	0.584	0.820	0.925	17.03	0.780	0.903	0.997	20.25	0.888	0.905	1.163	19.15	0.849	0.912	1.035	20.44	0.910	0.902	1.146
Cucumber	35	22.58	0.879	0.919	1.304	22.73	0.725	0.862	1.059	22.70	0.834	0.927	1.157	21.63	0.837	0.931	1.409	21.34	0.817	0.955	1.310	23.24	0.865	0.929	1.321
Lady Finger	36	20.66	0.874	0.925	1.189	21.42	0.721	0.876	0.878	21.69	0.848	0.926	0.990	21.64	0.882	0.928	1.246	21.50	0.866	0.954	1.110	23.04	0.903	0.926	1.148
Lemon	39	19.15	0.877	0.873	0.899	14.47	0.551	0.766	0.786	15.37	0.697	0.873	0.816	20.43	0.878	0.875	0.978	21.13	0.894	0.895	0.841	21.20	0.901	0.876	0.962
Orange	29	19.27	0.905	0.873	0.896	13.98	0.536	0.784	0.793	15.16	0.696	0.894	0.788	19.49	0.895	0.899	0.989	21.44	0.915	0.905	0.816	21.06	0.909	0.898	0.991
Peach	27	19.99	0.841	0.849	0.882	16.09	0.550	0.789	0.820	18.63	0.797	0.876	0.811	19.07	0.857	0.902	1.010	19.39	0.833	0.878	0.739	19.53	0.861	0.901	0.997
Pear	28	19.60	0.883	0.926	0.885	16.72	0.591	0.835	0.831	18.59	0.796	0.932	0.888	23.13	0.917	0.943	1.057	21.14	0.893	0.952	0.915	23.87	0.935	0.940	1.026
Plum	35	21.48	0.726	0.888	1.212	22.14	0.626	0.836	1.063	23.43	0.768	0.898	1.114	21.87	0.747	0.901	1.310	20.59	0.734	0.912	1.195	23.37	0.764	0.901	1.263
Pomegranate	55	19.47	0.875	0.901	1.285	16.44	0.602	0.843	1.112	17.56	0.740	0.915	1.165	19.25	0.853	0.925	1.338	19.92	0.863	0.949	1.240	18.72	0.817	0.922	1.295
Potato	35	24.48	0.939	0.862	1.028	18.83	0.622	0.766	0.869	21.81	0.854	0.862	0.945	22.43	0.935	0.860	1.102	23.53	0.935	0.892	1.022	23.19	0.946	0.857	1.046
Strawberry	35	20.84	0.920	0.955	1.444	15.48	0.621	0.880	1.157	15.96	0.743	0.932	1.222	19.13	0.881	0.932	1.485	19.59	0.891	0.972	1.440	20.24	0.898	0.920	1.466
Tomato	48	18.66	0.883	0.893	0.943	15.43	0.584	0.819	0.815	17.44	0.777	0.918	0.858	19.49	0.876	0.918	1.043	20.04	0.878	0.924	0.901	21.30	0.895	0.913	1.036
Dataset-level	735	20.48	0.863	0.906	1.082	18.29	0.632	0.834	0.917	19.55	0.792	0.914	0.974	20.93	0.865	0.919	1.172	21.12	0.863	0.931	1.045	21.93	0.881	0.916	1.135

colorization techniques. Fig. 10 shows the images from each category from our Natural-Color Dataset (NCD).

Lacking of Competitions: Currently, competitions for most vision tasks are held across various top-tier conferences, such as CVPR, ECCV workshops like NTIRE, PBVS, etc., and online submission platforms like Kaggle. These competitions help push the state-of-the-art. Unfortunately, there is no such arrangement for image colorization. Making

competitions a regular feature in top-tier conferences and online venues would thus be a drastic step forward for image colorization.

Limited Availability of Open-Source Codes: Open-source code plays a vital role in the advancement of the research fields, as can be seen in classification [64], image super-resolution [106], image denoising [107], etc. In image colorization, open-source codes are rare, or codes are obsolete now as they were built in earlier CNN frameworks. In other research fields, volunteers recycle or re-implement the codes for

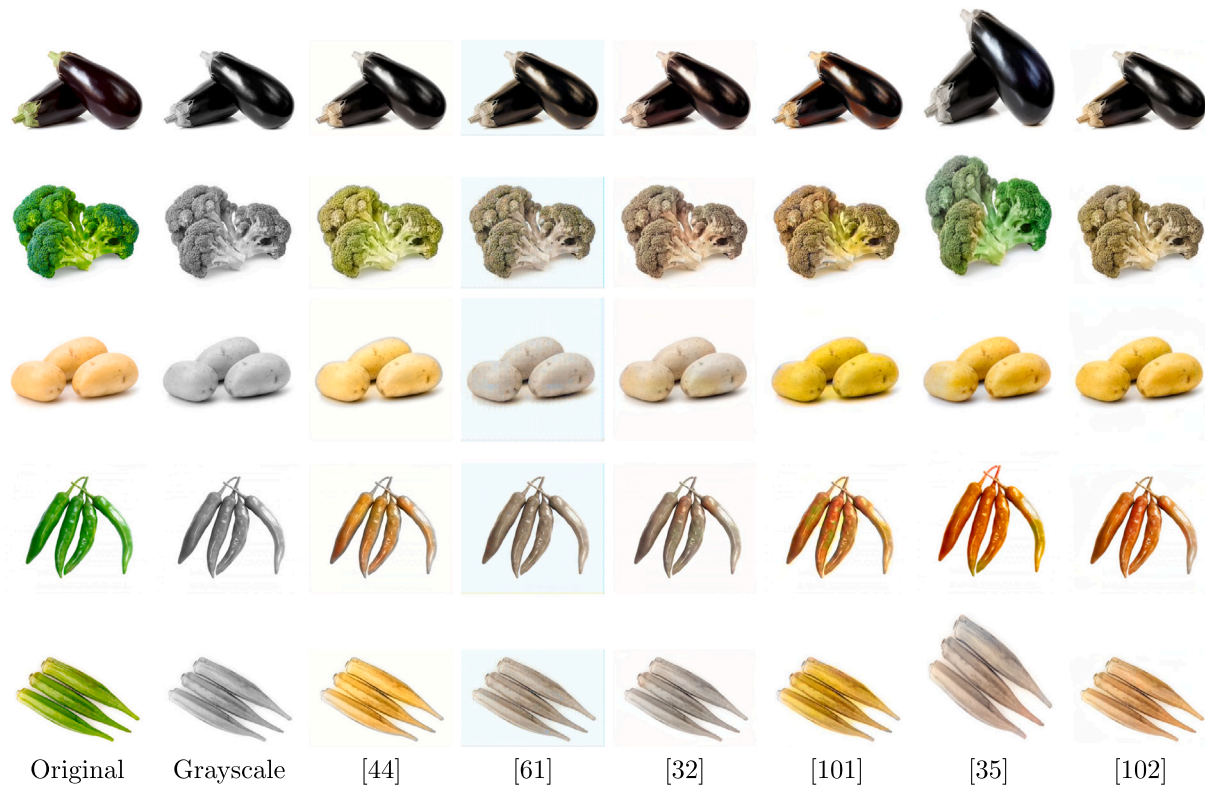


Fig. 12. Qualitative comparison on a few sample images of vegetables from Natural-Color Dataset. Most of the algorithms fail to reproduce the original colors.

the new frameworks and environments. However, this is not currently done for image colorization, hindering progress.

Long-Range Dependency & Multi-Scale Feature Fusion: In image colorization, distant objects and features are usually correlated in terms of context and color. Long-range dependencies can model such relationships. Likewise, image self-attention mechanisms [108] can also be used to predict the colors of distant objects in complex images by correlating different areas of the entire image. Dependency modeling guarantees color consistency across numerous regions of an image. Furthermore, the global structures detected from a low-resolution image and fine details extracted from a high-resolution version of the same image can be fused using the concept of multi-scale feature fusion [109] to exploit various levels of information for complete image understanding. The image colorization algorithms can benefit from the collective application of long-range dependency modeling and multi-scale feature fusion. Nevertheless, the researchers have not developed such algorithms to address the image colorization problem due to the resource-intensive nature and lack of real-time processing capability. Similarly, the fusion of features at various scales may result in information loss. Hence, careful consideration and design of the colorization model are required. Researchers in image colorization may explore and investigate the applicability of long-range dependency modeling and multi-scale feature fusion for image colorization tasks.

6. Conclusion

Single image colorization is a research problem with critical real-life applications. The exceptional success of deep learning approaches has led to a rapid growth in deep convolutional techniques for image colorization. Based on exciting innovations, various methods are proposed for exploiting network structures, training methods, learning paradigms, etc. This article presents a thorough review of deep-learning methods for image colorization.

We observe that image colorization performance has improved in recent years at the cost of increasing network complexity. However, inadequate metrics, network complexity, and failure to handle real-life degradations restrict the application of state-of-the-art methods to critical real-world scenarios.

We identified the following trends in image colorization: (1) GAN-based methods deliver diverse colorization visually compared to CNN-based methods; (2) the existing models typically produce a suboptimal result for complex scenes having a large number of objects with small sizes; (3) deep models with higher complexity have slight improvement in terms of numbers; (4) the diversity of networks in image colorization as compared to other image restoration is significant; (5) a future direction for image colorization is unsupervised learning; (6) many recent advancements and techniques such as attention mechanisms and loss functions can be incorporated for performance. We believe this article and our novel dataset will inspire additional efforts to address the aforementioned critical issues.

CRedit authorship contribution statement

Saeed Anwar: Writing – review & editing, Writing – original draft, Visualization, Validation, Supervision, Resources, Project administration, Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **Muhammad Tahir:** Writing – review & editing, Writing – original draft, Visualization, Validation, Resources, Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **Chongyi Li:** Writing – original draft, Validation, Resources, Methodology, Investigation. **Ajmal Mian:** Writing – review & editing, Writing – original draft, Validation, Supervision, Methodology, Formal analysis, Data curation. **Fahad Shahbaz Khan:** Writing – review & editing, Writing – original draft, Supervision. **Abdul Wahab Muzaffar:** Writing – review & editing, Writing – original draft, Validation, Investigation, Data curation.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data is shared on github.

References

- [1] Z. Cheng, Q. Yang, B. Sheng, Deep colorization, in: IEEE International Conference on Computer Vision, 2015, pp. 415–423.
- [2] S. Yoo, H. Bahng, S. Chung, J. Lee, J. Chang, J. Choo, Coloring with limited data: Few-shot colorization via memory augmented networks, in: IEEE Conference on Computer Vision and Pattern Recognition, 2019, pp. 11283–11292.
- [3] T. Welsh, M. Ashikhmin, K. Mueller, Transferring color to greyscale images, in: 29th Annual Conference on Computer Graphics and Interactive Techniques, 2002, pp. 277–280.
- [4] A. Levin, D. Lischinski, Y. Weiss, Colorization using optimization, in: Proceedings of International Conference on Computer Graphics and Interactive Techniques' ACM, 2004, pp. 689–694.
- [5] Y.-C. Huang, Y.-S. Tung, J.-C. Chen, S.-W. Wang, J.-L. Wu, An adaptive edge detection based colorization algorithm and its applications, in: 13th Annual ACM International Conference on Multimedia, 2005, pp. 351–354.
- [6] Y. Qu, T.-T. Wong, P.-A. Heng, Manga colorization, *ACM Trans. Graph.* 25 (3) (2006) 1214–1220.
- [7] L. Yatziv, G. Sapiro, Fast image and video colorization using chrominance blending, *IEEE Trans. Image Process.* 15 (5) (2006) 1120–1129.
- [8] Q. Luan, F. Wen, D. Cohen-Or, L. Liang, Y.-Q. Xu, H.-Y. Shum, Natural image colorization, in: 18th Eurographics Conference on Rendering Techniques, 2007, pp. 309–320.
- [9] T. Pärnmaa, L. Parts, Accurate classification of protein subcellular localization from high-throughput microscopy images using deep learning, *G3: Genes Genomes Genet.* 7 (5) (2017) 1385–1392.
- [10] M. Xiao, X. Shen, W. Pan, Application of deep convolutional neural networks in classification of protein subcellular localization with microscopy images, *Genet. Epidemiol.* (2019).
- [11] T. Young, D. Hazarika, S. Poria, E. Cambria, Recent trends in deep learning based natural language processing [review article], *IEEE Comput. Intell. Mag.* 13 (3) (2018) 55–75.
- [12] Z. Zhao, P. Zheng, S. Xu, X. Wu, Object detection with deep learning: A review, *IEEE Trans. Neural Netw. Learn. Syst.* 30 (11) (2019) 3212–3232.
- [13] D. Liu, L.T. Yang, P. Wang, R. Zhao, Q. Zhang, TT-TSVD: A multi-modal tensor train decomposition with its application in convolutional neural networks for smart healthcare, *ACM Trans. Multimedia Comput. Commun. Appl. (TOMM)* 18 (1s) (2022) 1–17.
- [14] P. Wang, L.T. Yang, G. Qian, F. Lu, The cyber-physical-social transition tensor service framework, *IEEE Trans. Sustain. Comput.* 6 (3) (2018) 481–492.
- [15] P. Wang, L.T. Yang, J. Li, X. Li, X. Zhou, MMDP: A mobile-IoT based multi-modal reinforcement learning service framework, *IEEE Trans. Serv. Comput.* 13 (4) (2020) 675–684.
- [16] Q. Jiang, J. Huang, X. Jin, P. Wang, W. Zhou, S. Yao, Medical image fusion using a new entropy measure between intuitionistic fuzzy sets joint Gaussian curvature filter, *IEEE Trans. Radiat. Plasma Med. Sci.* 7 (5) (2023) 494–508.
- [17] S. Anwar, S. Khan, N. Barnes, A deep journey into super-resolution: A survey, *ACM Comput. Surv.* 53 (3) (2020) <http://dx.doi.org/10.1145/3390462>.
- [18] N. Aafaq, A. Mian, W. Liu, S.Z. Gilani, M. Shah, Video description: A survey of methods, datasets, and evaluation metrics, *ACM Comput. Surv.* 52 (6) (2019) <http://dx.doi.org/10.1145/3355390>.
- [19] R. Zhang, P. Isola, A.A. Efros, Colorful image colorization, in: European Conference on Computer Vision, Springer, 2016, pp. 649–666.
- [20] F.M. Carlucci, P. Russo, B. Caputo, (DE)² CO: Deep depth colorization, *IEEE Robot. Autom. Lett.* 3 (3) (2018) 2386–2393.
- [21] Z. Hu, O. Shkurat, M. Kasner, Grayscale image colorization method based on U-net network, *Int. J. Image Graph. Signal Process.* 16 (2024) 70–82.
- [22] P. Sangkloy, J. Lu, C. Fang, F. Yu, J. Hays, Scribbler: Controlling deep image synthesis with sketch and color, in: IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 5400–5409.
- [23] R. Zhang, J.-Y. Zhu, P. Isola, X. Geng, A.S. Lin, T. Yu, A.A. Efros, Real-time user-guided image colorization with learned deep priors, *ACM Trans. Graph.* 36 (4) (2017) 1–11.
- [24] Y. Xiao, P. Zhou, Y. Zheng, C.-S. Leung, Interactive deep colorization using simultaneous global and local inputs, in: ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP, 2019, pp. 1887–1891.
- [25] Y. Ci, X. Ma, Z. Wang, H. Li, Z. Luo, User-guided deep anime line art colorization with conditional adversarial networks, in: 26th ACM International Conference on Multimedia, 2018, pp. 1536–1544.
- [26] M. Limmer, H.P. Lensch, Infrared colorization using deep convolutional neural networks, in: 2016 15th IEEE International Conference on Machine Learning and Applications, ICMLA, 2016, pp. 61–68.
- [27] P. Wang, V.M. Patel, Generating high quality visible images from SAR images using CNNs, in: 2018 IEEE Radar Conference (RadarConf18), 2018, pp. 0570–0575.
- [28] Q. Song, F. Xu, Y.-Q. Jin, Radar image colorization: Converting single-polarization to fully polarimetric using deep neural networks, *IEEE Access* 6 (2017) 1647–1661.
- [29] L. Junsoo, K. Eungyeup, L. Yunsung, K. Dongjun, C. Jaehyuk, C. Jaegul, Reference-based sketch image colorization using augmented-self reference and dense semantic correspondence, in: IEEE Conference on Computer Vision and Pattern Recognition, 2020, pp. 5801–5810.
- [30] V. Manjunatha, M. Iyyer, J. Boyd-Graber, L. Davis, Learning to color from language, in: 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers), 2018, pp. 764–769.
- [31] H. Bahng, S. Yoo, W. Cho, D. Keetae Park, Z. Wu, X. Ma, J. Choo, Coloring with words: Guiding image colorization through text-based palette generation, in: IEEE European Conference on Computer Vision, ECCV, 2018, pp. 431–447.
- [32] W.-T. Chu, Y.-T. Hsu, Depth-aware image colorization network, in: Proceedings of the 2018 Workshop on Understanding Subjective Attributes of Data, with the Focus on Evoked Emotions, 2018, pp. 17–23.
- [33] Y. Cao, Z. Zhou, W. Zhang, Y. Yu, Unsupervised diverse colorization via generative adversarial networks, in: Joint European Conference on Machine Learning and Knowledge Discovery in Databases, Springer, 2017, pp. 151–166.
- [34] K. Frans, Outline colorization through tandem adversarial networks, 2017, arXiv preprint arXiv:1704.08834.
- [35] K. Nazeri, E. Ng, M. Ebrahimi, Image colorization using generative adversarial networks, in: International Conference on Articulated Motion and Deformable Objects, Springer, 2018, pp. 85–94.
- [36] A. Deshpande, J. Lu, M.-C. Yeh, M. Jin Chong, D. Forsyth, Learning diverse image colorization, in: IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 6837–6845.
- [37] P. Vitoria, L. Raad, C. Ballester, ChromaGAN: Adversarial picture colorization with semantic class distribution, in: The IEEE Winter Conference on Applications of Computer Vision, 2020, pp. 2445–2454.
- [38] B. Li, Y. Lu, W. Pang, H. Xu, Image Colorization using CycleGAN with semantic and spatial rationality, *Multimedia Tools Appl.* 82 (14) (2023) 21641–21655.
- [39] H. Shafiq, B. Lee, Transforming color: A novel image colorization method, *Electronics* 13 (13) (2024) 2511.
- [40] Y. Wu, X. Wang, Y. Li, H. Zhang, X. Zhao, Y. Shan, Towards vivid and diverse image colorization with generative color prior, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2021, pp. 14377–14386.
- [41] S. Iizuka, E. Simo-Serra, H. Ishikawa, Let there be color! Joint end-to-end learning of global and local image priors for automatic image colorization with simultaneous classification, *ACM Trans. Graph. (ToG)* 35 (4) (2016) 1–11.
- [42] G. Larsson, M. Maire, G. Shakhnarovich, Learning representations for automatic colorization, in: European Conference on Computer Vision, Springer, 2016, pp. 577–593.
- [43] S. Guadarrama, R. Dahl, D. Bieber, M. Norouzi, J. Shlens, K. Murphy, PixColor: Pixel recursive colorization, in: 28th British Machine Vision Conference (BMVC), 2017.
- [44] G. Ozbulak, Image colorization by capsule networks, in: IEEE Conference on Computer Vision and Pattern Recognition Workshops, 2019.
- [45] J. Zhao, J. Han, L. Shao, C.G. Snoek, Pixelated semantic colorization, *Int. J. Comput. Vis.* (2019) 1–17.
- [46] M.H. Baig, L. Torresani, Multiple hypothesis colorization and its application to image compression, *Comput. Vis. Image Underst.* 164 (2017) 111–123.
- [47] J. Zhao, L. Liu, C. Snoek, J. Han, L. Shao, Pixel-level semantics guided image colorization, in: British Machine Vision Conference 2018, BMVC 2018, Northumbria University, Newcastle, UK, September 3-6, 2018, BMVA Press, 2018, p. 156, URL <http://bmvc2018.org/contents/papers/0236.pdf>.
- [48] X. Duan, Y. Cao, R. Zhang, X. Wang, P. Li, Shadow-aware image colorization, *Vis. Comput.* 40 (2024) 4969–4979.
- [49] M. He, D. Chen, J. Liao, P.V. Sander, L. Yuan, Deep exemplar-based colorization, *ACM Trans. Graph.* 37 (4) (2018) 47.
- [50] S. Jheng-Wei, C. Hung-Kuo, H. Jia-Bin, Instance-aware image colorization, in: IEEE Conference on Computer Vision and Pattern Recognition, 2020, pp. 7968–7977.
- [51] X. Zhongyou, W. Tingting, F. Faming, S. Yun, Z. Guixu, Stylization-based architecture for fast deep exemplar colorization, in: IEEE Conference on Computer Vision and Pattern Recognition, 2020, pp. 9363–9372.
- [52] Y. Wang, W.Q. Yan, Colorizing grayscale CT images of human lungs using deep learning methods, *Multimedia Tools Appl.* 81 (26) (2022) 37805–37819.
- [53] G. Petschnigg, R. Szeliski, M. Agrawala, M. Cohen, H. Hoppe, K. Toyama, Digital photography with flash and no-flash image pairs, *ACM Trans. Graph. (TOG)* 23 (3) (2004) 664–672.

- [54] E. Tola, V. Lepetit, P. Fua, A fast local descriptor for dense matching, in: 2008 IEEE Conference on Computer Vision and Pattern Recognition, 2008, pp. 1–8.
- [55] J. Xiao, J. Hays, K.A. Ehinger, A. Oliva, A. Torralba, Sun database: Large-scale scene recognition from abby to zoo, in: 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2010, pp. 3485–3492.
- [56] S. Ioffe, C. Szegedy, Batch normalization: Accelerating deep network training by reducing internal covariate shift, in: International Conference on Machine Learning, 2015, pp. 448–456.
- [57] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, T. Darrell, Caffe: Convolutional architecture for fast feature embedding, in: 22nd ACM International Conference on Multimedia, 2014, pp. 675–678.
- [58] D.P. Kingma, J. Ba, Adam: A method for stochastic optimization, in: International Conference on Learning Representations, 2015.
- [59] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, L. Fei-Fei, Imagenet: A large-scale hierarchical image database, in: IEEE Conference on Computer Vision and Pattern Recognition, 2009, pp. 248–255.
- [60] A.L. Maas, A.Y. Hannun, A.Y. Ng, Rectifier nonlinearities improve neural network acoustic models, in: Proc. Icml, Vol. 30, 2013, p. 3.
- [61] A. Krizhevsky, I. Sutskever, G.E. Hinton, Imagenet classification with deep convolutional neural networks, in: Advances in Neural Information Processing Systems, 2012, pp. 1097–1105.
- [62] K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition, in: International Conference on Learning Representations, 2015.
- [63] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. Rabinovich, Going deeper with convolutions, in: 2015 IEEE Conference on Computer Vision and Pattern Recognition, CVPR, 2015, pp. 1–9.
- [64] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 770–778.
- [65] K. Lai, L. Bo, X. Ren, D. Fox, A large-scale hierarchical multi-view rgb-d object dataset, in: 2011 IEEE International Conference on Robotics and Automation, 2011, pp. 1817–1824.
- [66] C. Li, A. Reiter, G.D. Hager, Beyond spatial pooling: fine-grained representation learning in multiple domains, in: IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 4913–4922.
- [67] A. Singh, J. Sha, K.S. Narayan, T. Achim, P. Abbeel, Bigbird: A large-scale 3d database of object instances, in: 2014 IEEE International Conference on Robotics and Automation, ICRA, 2014, pp. 509–516.
- [68] Y. Güçlütürk, U. Güçlü, R. van Lier, M.A. van Gerven, Convolutional sketch inversion, in: European Conference on Computer Vision, Springer, 2016, pp. 810–824.
- [69] O. Ronneberger, P. Fischer, T. Brox, U-net: Convolutional networks for biomedical image segmentation, in: International Conference on Medical Image Computing and Computer-Assisted Intervention, Springer, 2015, pp. 234–241.
- [70] B. Zhou, A. Lapedriza, A. Torralba, A. Oliva, Places: An image database for deep scene understanding, *J. Vis.* 17 (10) (2017) 296.
- [71] M. Mirza, S. Osindero, Conditional generative adversarial nets, in: Advances in Neural Information Processing Systems, 2014, pp. s 2672–2680.
- [72] M. Saito, Y. Matsui, Illustration2vec: a semantic vector representation of illustrations, in: SIGGRAPH Asia 2015 Technical Briefs, 2015, pp. 1–4.
- [73] S. Xie, R. Girshick, P. Dollár, Z. Tu, K. He, Aggregated residual transformations for deep neural networks, in: IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 1492–1500.
- [74] C. Ledig, L. Theis, F. Huszár, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang, et al., Photo-realistic single image super-resolution using a generative adversarial network, in: IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 4681–4690.
- [75] S.E. Reed, Z. Akata, S. Mohan, S. Tenka, B. Schiele, H. Lee, Learning what and where to draw, in: Advances in Neural Information Processing Systems, 2016, pp. 217–225.
- [76] E. Perez, F. Strub, H. De Vries, V. Dumoulin, A. Courville, Film: Visual reasoning with a general conditioning layer, in: Thirty-Second AAAI Conference on Artificial Intelligence, 2018.
- [77] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, C.L. Zitnick, Microsoft coco: Common objects in context, in: European Conference on Computer Vision, Springer, 2014, pp. 740–755.
- [78] A. Radford, L. Metz, S. Chintala, Unsupervised representation learning with deep convolutional generative adversarial networks, in: Proc. the International Conference on Learning Representations (ICLR), 2015.
- [79] F. Yu, A. Seff, Y. Zhang, S. Song, T. Funkhouser, J. Xiao, Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop, 2015, arXiv preprint arXiv:1506.03365.
- [80] J. Long, E. Shelhamer, T. Darrell, Fully convolutional networks for semantic segmentation, in: CVPR, 2015, pp. 3431–3440.
- [81] K. He, X. Zhang, S. Ren, J. Sun, Delving deep into rectifiers: Surpassing human-level performance on imagenet classification, in: IEEE International Conference on Computer Vision, 2015, pp. 1026–1034.
- [82] A. Krizhevsky, G. Hinton, et al., Learning Multiple Layers of Features from Tiny Images (Master's thesis), University of Tront, 2009.
- [83] P. Isola, J.-Y. Zhu, T. Zhou, A.A. Efros, Image-to-image translation with conditional adversarial networks, in: IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 1125–1134.
- [84] L. Kaiser, O. Nachum, A. Roy, S. Bengio, Learning to remember rare events, in: International Conference on Learning Representations, 2017.
- [85] M.-E. Nilsback, A. Zisserman, Automated flower classification over a large number of classes, in: 2008 Sixth Indian Conference on Computer Vision, Graphics & Image Processing, 2008, pp. 722–729.
- [86] P. Docter, J. Culton, J. Pidgeon, R. Eggleston, Monsters, Inc., Walt Disney Pictures, 2001.
- [87] M.D. Zeiler, Adadelta: an adaptive learning rate method, 2012, arXiv preprint arXiv:1212.5701.
- [88] L. Bottou, Stochastic gradient learning in neural networks, *Proc. Neuro-Nimes* 91 (8) (1991) 12.
- [89] R. Collobert, K. Kavukcuoglu, C. Farabet, Torch7: A matlab-like environment for machine learning, in: BigLearn, NIPS Workshop, (CONF) 2011.
- [90] S. Sabour, N. Frosst, G.E. Hinton, Dynamic routing between capsules, in: Advances in Neural Information Processing Systems, 2017, pp. 3856–3866.
- [91] E. Agustsson, R. Timofte, Ntire 2017 challenge on single image super-resolution: Dataset and study, in: IEEE Conference on Computer Vision and Pattern Recognition Workshops, 2017, pp. 126–135.
- [92] M. Everingham, L. Van Gool, C.K. Williams, J. Winn, A. Zisserman, The pascal visual object classes (voc) challenge, *Int. J. Comput. Vis.* 88 (2) (2010) 303–338.
- [93] J. Liao, Y. Yao, L. Yuan, G. Hua, S.B. Kang, Visual attribute transfer through deep image analogy, *ACM Trans. Graph.* 36 (4) (2017) 1–15.
- [94] H. Xun, B. Serge, Arbitrary style transfer in real-time with adaptive instance normalization, in: ICCV, 2017, pp. 1501–1510.
- [95] H. Peter, Robust estimation of a location parameter, in: Breakthroughs in Statistics, 1992, pp. 492–518.
- [96] H. Caesar, J. Uijlings, V. Ferrari, Coco-stuff: Thing and stuff classes in context, in: IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 1209–1218.
- [97] Z. Bolei, L. Agata, X. Jianxiang, T. Antonio, O. Aude, Learning deep features for scene recognition using places dataset, in: Advances in Neural Information Processing Systems, 2014, pp. 487–495.
- [98] B. Zhou, A. Lapedriza, J. Xiao, A. Torralba, A. Oliva, Learning deep features for scene recognition using places database, in: NIPS, 2014.
- [99] M. Everingham, S.A. Eslami, L. Van Gool, C.K. Williams, J. Winn, A. Zisserman, The pascal visual object classes challenge: A retrospective, *Int. J. Comput. Vis.* 111 (1) (2015) 98–136.
- [100] A. Torralba, R. Fergus, W.T. Freeman, 80 Million tiny images: A large data set for nonparametric object and scene recognition, *IEEE Trans. Pattern Anal. Mach. Intell.* 30 (11) (2008) 1958–1970.
- [101] Z. Wang, A.C. Bovik, H.R. Sheikh, E.P. Simoncelli, Image quality assessment: from error visibility to structural similarity, *IEEE Trans. Image Process.* (2004).
- [102] J. Antic, A deep learning based project for colorizing and restoring old images, 2018.
- [103] C. Lei, Q. Chen, Fully automatic video colorization with self-regularization and diversity, in: CVPR, 2019, pp. 3753–3761.
- [104] S. Wang, K. Ma, H. Yeganeh, Z. Wang, W. Lin, A patch-structure representation method for quality assessment of contrast changed images, *IEEE Signal Process. Lett.* 22 (12) (2015) 2387–2390.
- [105] K. Panetta, C. Gao, S. Agaian, Human-visual-system-inspired underwater image quality measures, *IEEE J. Ocean. Eng.* 41 (3) (2015) 541–551.
- [106] S. Anwar, N. Barnes, Densely residual laplacian super-resolution, *IEEE Trans. Pattern Anal. Mach. Intell.* (2020).
- [107] S. Anwar, N. Barnes, Real image denoising with feature attention, in: IEEE International Conference on Computer Vision, 2019, pp. 3155–3164.
- [108] X. Qin, M. Li, Y. Liu, H. Zheng, J. Chen, M. Zhang, An efficient coding-based grayscale image automatic colorization method combined with attention mechanism, *IET Image Process.* 16 (7) (2022) 1765–1777.
- [109] L. Zhou, S. Zhao, Z. Wan, Y. Liu, Y. Wang, X. Zuo, MFEFNet: A multi-scale feature information extraction and fusion network for multi-scale object detection in UAV aerial images, *Drones* 8 (5) (2024) 186.