

Deep Colorization

Zezhou Cheng, *Student Member, IEEE*, Qingxiong Yang, *Member, IEEE*, Bin Sheng, *Member, IEEE*,

<http://www.cs.cityu.edu.hk/~qiyang/publications/iccv15/>

Abstract—This paper investigates into the **colorization problem** which converts a grayscale image to a colorful version. This is a very difficult problem and normally requires manual adjustment to achieve artifact-free quality. For instance, it normally requires human-labelled color scribbles on the grayscale target image or a careful selection of colorful reference images (e.g., capturing the same scene in the grayscale target image). Unlike the previous methods, this paper aims at a **high-quality fully-automatic colorization method**. With the **assumption of a perfect patch matching technique**, the use of an extremely large-scale reference database (that contains sufficient color images) is the most reliable solution to the colorization problem. However, **patch matching noise will increase with respect to the size of the reference database in practice**. Inspired by the recent success in deep learning techniques which provide amazing modeling of large-scale data, this paper re-formulates the colorization problem so that **deep learning techniques can be directly employed**. To ensure **artifact-free quality**, a **joint bilateral filtering based post-processing step** is proposed. We further develop an **adaptive image clustering technique** to incorporate the **global image information**. Numerous experiments demonstrate that our method outperforms the state-of-art algorithms both in terms of quality and speed.

Index Terms—Image colorization, neural networks

I. INTRODUCTION

Image colorization assigns a color to each pixel of a target grayscale image. Colorization methods can be roughly divided into two categories: scribble-based colorization [2], [3], [4], [5], [6] and example-based colorization [7], [8], [9], [10], [11], [12]. The scribble-based methods typically require substantial efforts from the user to provide considerable scribbles on the target grayscale images. It is thus time-assuming to colorize a grayscale image with fine-scale structures, especially for a rookie user.

To reduce the burden on user, [12] proposes an example-based method which is later further improved by [7], [10]. The

example-based method typically transfers the color information from a similar reference image to the target grayscale image. However, finding a suitable reference image becomes an obstacle for a user. [8], [11] simplify this problem by utilizing the image data on the Internet and propose filtering schemes to select suitable reference images. However, they both have additional constraints. [11] requires identical Internet object for precise per-pixel registration between the reference images and the target grayscale image. It is thus limited to objects with a rigid shape (e.g. landmarks). [8] requires user to provide a semantic text label and segmentation cues for the foreground object. In practice, manual segmentation cues are hard to obtain as the target grayscale image may contain multiple complex objects (e.g. building, car, tree, elephant). These methods share the same limitation – their performance highly depends on the selected reference image(s).

A fully-automatic colorization method is proposed to address this limitation. Intuitively, one reference image cannot include all possible scenarios in the target grayscale image. As a result, [7], [8], [10], [12] require similar reference image(s). A more reliable solution is locating the most similar image patch/pixel in a huge reference image database and then transferring color information from the matched patch/pixel to the target patch/pixel. However, the matching noise is too high when a large-scale database is adopted in practice.

Deep learning techniques have achieved amazing success in modeling large-scale data recently. It has shown powerful learning ability that even outperforms human beings to some extent (e.g. [13]) and deep learning techniques have been demonstrated to be very effective on various computer vision and image processing applications including image classification [14], pedestrian detection [15], [16], image super-resolution [17], photo adjustment [18] etc. The success of deep learning techniques motivates us to explore its potential application in our context. This paper formulates image colorization as a regression problem and deep neural networks are used to solve the problem. A large database of reference images comprising all kinds of objects (e.g. tree, building, sea, mountain etc.) is used for training the neural networks. Some example reference images are presented in Figure 1 (a). Although the training is significantly slow due to the adoption of a large database, the learned model can be directly used to colorize a target grayscale image efficiently. The state-of-the-art colorization methods normally require matching between the target and reference images and thus are slow.

It has recently been demonstrated that high-level understanding of an image is of great use for low-level vision problems (e.g. image enhancement [18], edge detection [19]). Because image colorization is typically semantic-aware, we propose a new semantic feature descriptor to incorporate the

Preliminary version of this work was published in ICCV 2015 [1]

Zezhou Cheng is with the Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai 200240, China (e-mail: chengzezhou@sjtu.edu.cn).

Qingxiong Yang is with the Department of Computer Science at the City University of Hong Kong, Hong Kong, China (e-mail: qiyang@cityu.edu.hk).

Bin Sheng is with the same Department as Zezhou Cheng (e-mail: shengbin@sjtu.edu.cn).

Matlab code, trained models and more colorization results are available at the authors' website.

semantic-awareness into our colorization model.

An adaptive image clustering is proposed to incorporate the global image information to reduce the training ambiguities.

To demonstrate the effectiveness of the presented approach, we train our deep neural networks using a large set of reference images from different categories as can be seen in Figure 1 (a). The learned model is then used to colorize various grayscale images in Figure 14. The colorization results shown in Figure 14 demonstrate the robustness and effectiveness of the proposed method.

The major contributions of this paper are as follows:

- 1) It proposes the first deep learning based image colorization method and demonstrates its effectiveness on various scenes.
- 2) It carefully analyzes informative yet discriminative image feature descriptors from low to high level, which is key to the success of the proposed colorization method.

An initial version of this work was presented in [1]. The present work has significant differences from the earlier version. Firstly, we propose an adaptive image clustering to classify the training images according to their global information. A neural network is trained for each image cluster and the resulted neural network assemble is used to colorize the target grayscale image. Considerable qualitative and quantitative results are shown to prove that the new framework outperforms [1] both in colorization quality and accuracy. Secondly, more analysis of the proposed model along with comparisons to the state-of-art concurrent work [20] is added. Thirdly, we show that the proposed model is flexible to learn various colorization styles. Additionally, we update the experimental results reported in [1] due to changes between the preliminary and the current work.

II. RELATED WORK

This section gives a brief overview of the previous colorization methods.

Scribble-based colorization Levin et al. [3] propose an effective approach that requires the user to provide colorful scribbles on the grayscale target image. The color information on the scribbles are then propagated to the rest of the target image using least-square optimization. Huang et al. [2] develop an adaptive edge detection algorithm to reduce the color bleeding artifact around the region boundaries. Yatziv et al. [6] colorize the pixels using a weighted combination of user scribbles. Qu et al. [5] and Luan et al. [4] utilize the texture feature to reduce the amount of required scribbles.

Example-based colorization Unlike scribble-based colorization methods, the example-based methods transfer the color information from a reference image to the target grayscale image. The example-based colorization methods can be further divided into two categories according to the source of reference images:

- (1) Colorization using user-supplied example(s). This type of methods requires the user to provide a suitable reference image. Inspired by image analogies [21] and the color transfer technique [22], Welsh et al. [12] employ the pixel intensity and neighborhood statistics to find a similar pixel in the reference

image and then transfer the color of the matched pixel to the target pixel. It is later improved in [10] by taking into account the texture feature. Charpiat et al. [7] propose a global optimization algorithm to colorize a pixel. Gupta et al. [9] develop an colorization method based on superpixel to improve the spatial coherency. These methods share the limitation that the colorization quality relies heavily on example image(s) provided by the user. However, there is not a standard criteria on the example image(s), thus finding a suitable reference image is a difficult task.

(2) Colorization using web-supplied example(s). To release the users' burden of finding a suitable image, Liu et al. [11] and Chia et al. [8] utilize the massive image data on the Internet. Liu et al. [11] compute an intrinsic image using a set of similar reference images collected from the Internet. This method is robust to illumination difference between the target and reference images, but it requires the images to contain identical object(s)/scene(s) for precise per-pixel registration between the reference images and the target grayscale image. It is unable to colorize the dynamic factors (e.g. person, car) among the reference and target images, since these factors are excluded during the computation of the intrinsic image. As a result, it is limited to static scenes and the objects/scenes with a rigid shape (e.g. famous landmarks). Chia et al. [8] propose an image filter framework to distill suitable reference images from the collected Internet images. It requires the user to provide semantic text label to search for suitable reference image on the Internet and human-segmentation cues for the foreground objects.

More recently, Deshpande et al. [20] propose a learning based framework that formulates this problem as a quadratic objective function. Histogram correction is applied to improve the initial colorization results. However, a suitable scene histogram is required in their refinement step. The other limitation is their low speed of colorization.

In contrast to the previous colorization methods, the proposed method is fully automatic by involving a large set of reference images from different scenes (e.g., coast, highway, field etc.) with various objects (e.g., tree, car, building etc.) and performs with artifact-free quality and high speed.

III. OUR METRIC

An overview of the proposed colorization method is presented in Figure 1. Similar to the other learning based approaches, the proposed method has two major steps: (1) training a neural network assemble using a large set of example reference images; (2) using the learned neural network assemble to colorize a target grayscale image. These two steps are summarized in Algorithm 1 and 2, respectively.

A. A Deep Learning Model for Image Colorization

This section formulates image colorization as a regression problem and solves it using a regular deep neural network.

1) *Formulation*: A deep neural network is a universal approximator that can represent arbitrarily complex continuous functions [23]. Given a set of exemplars $\Lambda = \{\tilde{G}, \tilde{C}\}$, where \tilde{G} are grayscale images and \tilde{C} are corresponding color images

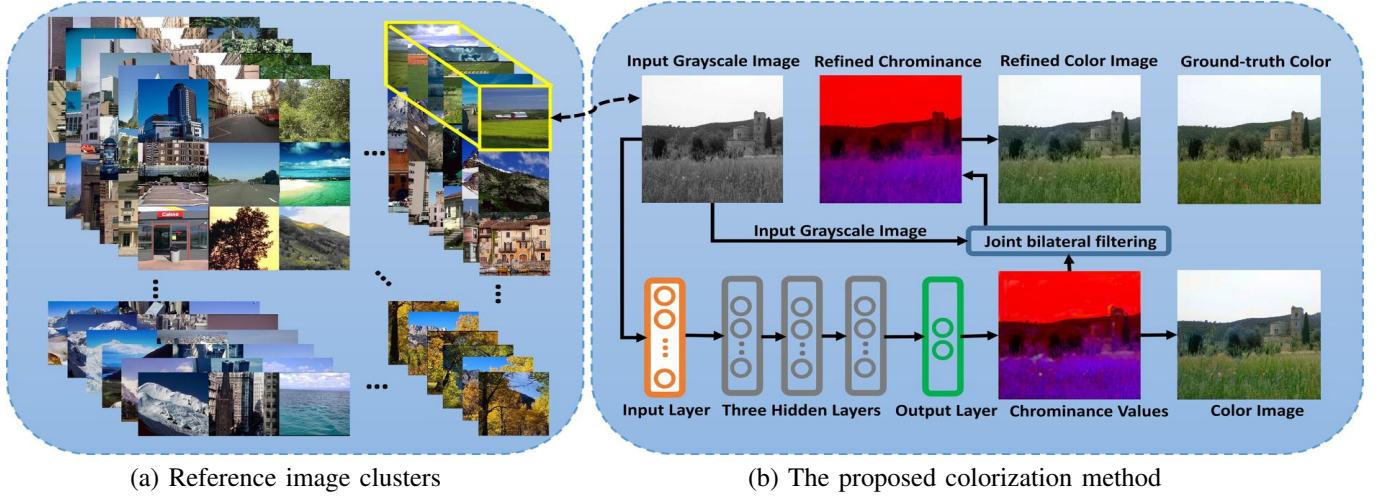


Fig. 1: The adopted large reference image database and overview of the proposed colorization method. (a) shows the reference images that have been grouped into various clusters by a proposed adaptive image clustering technique. A deep neural network (DNN) will be trained for each cluster. (b) presents our colorization procedure and the architecture of the proposed DNN. Given a target grayscale, the nearest cluster and corresponding trained DNN will be explored automatically first. The feature descriptors will be extracted at each pixel and serve as the input of the neural network. Each connection between pairs of neurons is associated with a weight to be learned from a large reference image database. The output is the chrominance of the corresponding pixel which can be directly combined with the luminance (grayscale pixel value) to obtain the corresponding color value. The chrominance computed from the trained model is likely to be a bit noisy around low-texture regions. The noise can be significantly reduced with a joint bilateral filter (with the input grayscale image as the guidance).

Algorithm 1 Image Colorization – Training Step

Input: Pairs of reference images: $\Lambda = \{\vec{G}, \vec{C}\}$.

Output: A trained neural network assemble.

- 1: Extract global descriptors of the reference images, group these images into different clusters adaptively and compute the semantic histogram of each cluster;
 - 2: Compute feature descriptors \vec{x} at sampled pixels in \vec{G} and the corresponding chrominance values \vec{y} in \vec{C} ;
 - 3: Construct a deep neural network for each cluster;
 - 4: Train the deep neural networks using the training set $\Psi = \{\vec{x}, \vec{y}\}$.
-

Algorithm 2 Image Colorization – Testing Step

Input: A target grayscale image I and the trained neural network assemble.

Output: A corresponding color image: \hat{I} .

- 1: Compute global descriptor and semantic histogram of I , then find its nearest cluster center and corresponding trained neural network;
 - 2: Extract a feature descriptor at each pixel location in I ;
 - 3: Send feature descriptors extracted from I to the trained neural network to obtain the corresponding chrominance values;
 - 4: Refine the chrominance values to remove potential artifacts;
 - 5: Combine the refined chrominance values and I to obtain the color image \hat{I} .
-

respectively, our method is based on a premise: there exists a complex gray-to-color mapping function \mathcal{F} that can map the features extracted at each pixel in \vec{G} to the corresponding chrominance values in \vec{C} . We aim at learning such a mapping function from Λ so that the conversion from a new gray image to color image can be achieved by using \mathcal{F} . In our model, the YUV color space is employed, since this color space minimizes the correlation between the three coordinate axes of the color space. For a pixel p in \vec{G} , the output of \mathcal{F} is simply the U and V channels of the corresponding pixel in \vec{C} and the input of \mathcal{F} is the feature descriptors we compute at pixel p . The feature descriptors are introduced in detail in Sec. III-B. We reformulate the gray-to-color mapping function as $c_p = \mathcal{F}(\Theta, x_p)$, where x_p is the feature descriptor extracted at pixel p and c_p are the corresponding chrominance values. Θ are the parameters of the mapping function \mathcal{F} to be learned from Λ .

We solve the following least squares minimization problem to learn the parameters Θ :

$$\argmin_{\Theta \in \Upsilon} \sum_{p=1}^n \|\mathcal{F}(\Theta, x_p) - c_p\|^2 \quad (1)$$

where n is the total number of training pixels sampled from Λ and Υ is the function space of $\mathcal{F}(\Theta, x_p)$.

2) *Architecture:* Deep neural networks (DNNs) typically consist of one input layer, multiple hidden layers and one output layer. Generally, each layer can comprise various numbers of neurons. In our model, the number of neurons in the input layer is equal to the dimension of the feature descriptor extracted from each pixel location in a grayscale image and the output layer has two neurons which output the U and V channels of the corresponding color value, respectively. We perceptually set the number of neurons in the hidden layer to

half of that in the input layer. Each neuron in the hidden or output layer is connected to all the neurons in the proceeding layer and each connection is associated with a weight. Let o_j^l denote the output of the j -th neuron in the l -th layer. o_j^l can be expressed as follows:

$$o_j^l = f(w_{j0}^l b + \sum_{i>0} w_{ji}^l o_i^{l-1}) \quad (2)$$

where w_{ji}^l is the weight of the connection between the j^{th} neuron in the l^{th} layer and the i^{th} neuron in the $(l-1)^{th}$ layer, the b is the bias neuron which outputs value one constantly and $f(z)$ is an activation function which is typically nonlinear (e.g., tanh, sigmoid, ReLU[14]). The output of the neurons in the output layer is just the weighted combination of the outputs of the neurons in the proceeding layer. In our method, we utilize ReLU[14] as the activation function as it can speed up the convergence of the training process. The architecture of our neural network is presented in Figure 1.

We apply the classical error back-propagation algorithm to train the connected power of the neural network, and the weights of the connections between pairs of neurons in the trained neural network are the parameters Θ to be learned.

B. Feature Descriptor

Feature design is key to the success of the proposed colorization method. There are massive candidate image features that may affect the effectiveness of the trained model (e.g. SIFT, SURF, Gabor, Location, Intensity histogram etc.). We conducted numerous experiments to test various features and kept only features that have practical impacts on the colorization results. We separate the adopted features into low-, mid- and high-level features. Let x_p^L , x_p^M , x_p^H denote different-level feature descriptors extracted from a pixel location p , we concatenate these features to construct our feature descriptor $x_p = \{x_p^L; x_p^M; x_p^H\}$. The adopted image features are discussed in detail in the following sections.

1) *Low-level Patch Feature*: Intuitively, there exist too many pixels with same luminance but fairly different chrominance in a color image, thus it's far from being enough to use only the luminance value to represent a pixel. In practice, different pixels typically have different neighbors, using a patch centered at a pixel p tends to be more robust to distinguish pixel p from other pixels in a grayscale image. Let x_p^L denote the array containing the sequential grayscale values in a 7×7 patch center at p , x_p^L is used as the low-level feature descriptor in our framework. This feature performs better than traditional features like SIFT and DAISY at low-texture regions when used for image colorization. Figure 2 shows the impact of patch feature on our model. Note that our model will be insensitive to the intensity variation within a semantic region when the patch feature is missing (e.g., the entire sea region is assigned with one color in Figure 2(b)).

2) *Mid-level DAISY Feature*: DAISY is a fast local descriptor for dense matching [24]. Unlike the low-level patch feature, DAISY can achieve a more accurate discriminative description of a local patch and thus can improve the colorization quality on complex scenarios. A DAISY descriptor is computed at

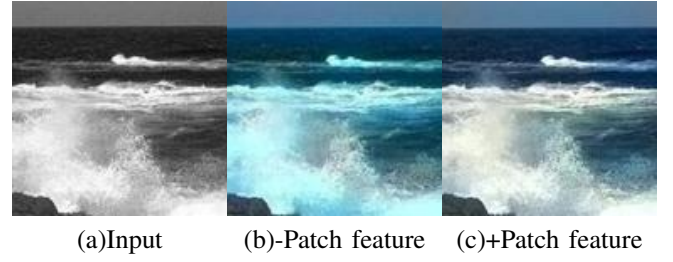


Fig. 2: Evaluation of patch feature. (a) is the target grayscale image. (b) removes the low-level patch feature and (c) includes all the proposed features.

a pixel location p in a grayscale image and is denote as x_p^M . Figure 3 demonstrates the performance with and without DAISY feature on a fine-structure object and presents the comparison with the state-of-the-art colorization methods. As can be seen, the adoption of DAISY feature in our model leads to a more detailed and accurate colorization result on complex regions. However, DAISY feature is not suitable for matching low-texture regions/objects and thus will reduce the performance around these regions as can be seen in Figure 3(c). A post-processing step will be introduced in Section III-B.4 to reduce the artifacts and its result is presented in Figure 3(d). Furthermore, we can see that our result is comparable to Liu et al. [11] (which requires a rigid-shape target object and identical reference objects) and Chia et al. [8] (which requires manual segmentation and identification of the foreground objects), although our method is fully-automatic.

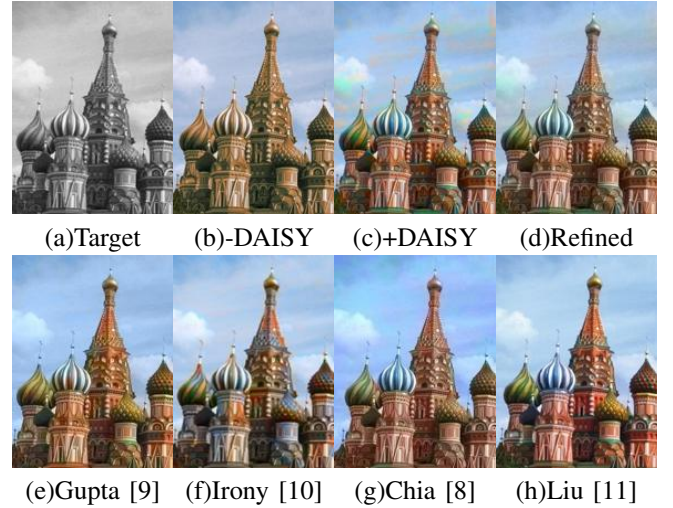


Fig. 3: Evaluation of DAISY feature. (a) is the target gray scale image. (b) is our result without DAISY feature. (c) is our result after incorporating DAISY feature into our model. (d) is the final result after artifact removal (see Sec. III-B.4 for details). (e)-(h) presents results obtained with the state-of-the-art colorizations. Although the proposed method is fully-automatic, its performance is comparable to the state-of-the-art.

3) *High-level Semantic Feature*: Patch and DAISY are low-level and mid-level features indicating the geometric structure of the neighbors of a pixel. The existing state-of-art methods typically employ such features to match pixels between the

reference and target images. Recently, high-level properties of an image have demonstrated its importance and virtues in some fields (e.g. image enhancement [18], edge detection [19]). Considering that the image colorization is typically a semantic-aware process, we extract a semantic feature at each pixel to express its category (e.g. sky, sea, animal) in our model.

We adopt the state-of-art scene parsing algorithm [25] to annotate each pixel with its category label, and obtain a semantic map for the input image. The semantic map is not accurate around region boundaries. As a result, it is smoothed using an efficient edge-preserving filter [26] with the guidance of the original gray scale image. An N-dimension probability vector will be computed at each pixel location, where N is the total number of object categories and each element is the probability that the current pixel belongs to the corresponding category. This probability vector is used as the high-level descriptor denoted as x_p^H .



Fig. 4: Importance of semantic feature. (a) is the target grayscale image. (b) is the colorization result using patch and DAISY features only. (c) is the result using patch, DAISY and semantic features.

Figure 4 shows that the colorization result may change significantly with and without the semantic feature. The adoption of semantic feature can significantly reduce matching/training ambiguities. For instance, if a pixel is detected to be inside a sky region, only sky color values residing in the reference image database will be used. The colorization problem is thus simplified after integrating the semantic information and colorization result is visually much better as can be seen in Figure 4.

4) *Chrominance Refinement*: The proposed method adopts the patch feature and DAISY feature, and we hope to use patch feature to describe low-texture simple regions and DAISY to describe fine-structure regions. However, we simply concatenate the two features instead of digging out a better combination. This will result in potential artifacts especially around the low-texture objects (e.g., sky, sea). This is because DAISY is vulnerable to these objects and presents a negative contribution.

The artifacts around low-texture regions can be significantly reduced using joint bilateral filtering technique [27]. It was first introduced to remove image noise of a no-flash image with the help of a noise-free flash image. Our problem is similar, the chrominance values obtained from the trained neural network is noisy (and thus results in visible artifacts) while the target grayscale image is noise-free. As a result, to ensure artifact-free quality, we apply joint bilateral filtering to smooth/refine the chrominance values (computed by the

trained neural network) with the target grayscale image as the guidance. Figure 5 presents the result before and after chrominance refinement. Note that most of the visible artifacts can be successfully removed.

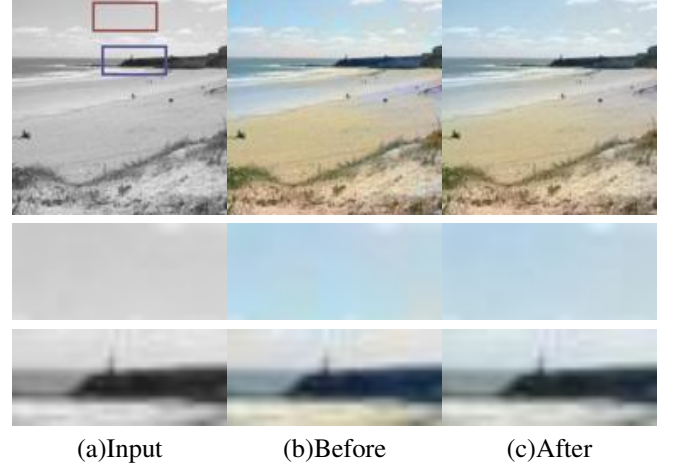


Fig. 5: Chrominance refinement using joint bilateral filtering [27]. From (a) to (c): target grayscale image, colorization results before and after chrominance refinement, respectively. Note that the artifacts in (b) are successfully removed from (c).

C. Adaptive Image Clustering

This section presents an adaptive image clustering technique and demonstrates its effectiveness in improving the colorization performance.

The proposed DNN trained from a large reference image set that contains various scenes performs well in most cases. However, visible artifacts still appear, especially on the objects with large color variances (e.g. building, plants etc.). One reason is that the receptive field of the DNN is limited on local patch, which causes large training ambiguities especially when large training set is utilized. Intuitively, the global image descriptor (e.g. gist [29], intensity histogram etc.) is able to reflect the scene category (e.g. coast, highway, city etc.) with the robustness to local noise, and there are typically smaller color variances within one scene than mixed scenes. Thus the global information is useful to reduce the matching/training ambiguities and improve the colorization accuracy. [1] reveals that feeding the global descriptor into DNN directly would produce an unnatural colorization result. In the present work, we incorporate the global information by an image clustering method. Inspired by [30] which adopts an adaptive pixel clustering algorithm and trains a regressor assemble to model the light transport, we utilize a similar strategy to split the reference images into different scenes, for each of which a DNN is trained.

As illustrated in Algorithm 3, the reference images are clustered adaptively on different layers by standard k-means clustering algorithm. After completing the training of DNN for cluster i on layer l , we measure the training error $E(I_{(i,l)})$ for each reference image $I_{(i,l)}$ as the negative Peak Signal-to-Noise Ratio (PSNR) computed from the colorization result

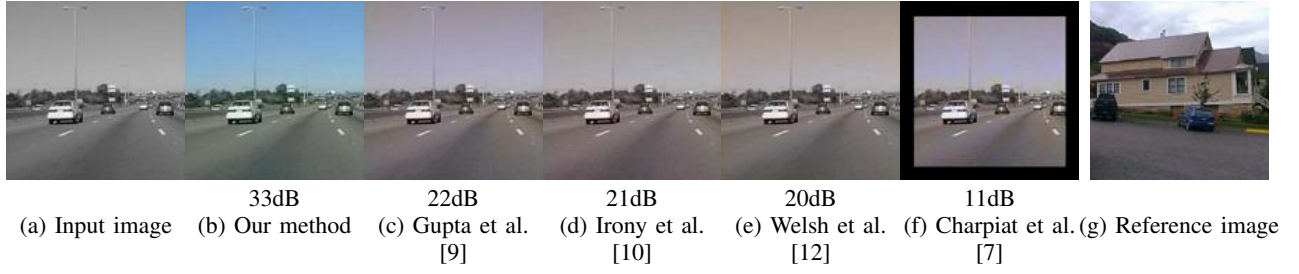


Fig. 6: Comparison with the state-of-art colorization methods [7], [9], [10], [12]. (c)-(f) use (g) as the reference image, while the proposed method adopts a large reference image dataset. The reference image contains similar objects as the target grayscale image (e.g., road, trees, building, cars). It is seen that the performance of the state-of-art colorization methods is lower than the proposed method when the reference image is not “optimal”. The segmentation masks used by [10] are computed by mean shift algorithm [28]. The PSNR values computed from the colorization results and ground truth are presented under the colorized images.

$\hat{I}_{(i,l)}$ and the ground truth image. If $E(I_{(i,l)})$ is lower than a threshold ε , $I_{(i,l)}$ will be removed from the reference image set $\Lambda_{(i,l)}$. As a result, the top layer contains all reference images while the lower layer comprises fewer images.

To ensure a sufficient number of samples for training a single DNN, the number of clusters on the next lower layer is determined by the size of Λ as well as the minimal number of reference images required for training a single DNN (denoted as μ). Similar to [30], we compute μ by the following equation according to [31]:

$$\mu = \frac{\alpha N_w}{N_s} \quad (3)$$

where α is a constant scale factor, N_w is the total number of weights in a single DNN, and N_s is the number of samples from one reference image.

Algorithm 3 Adaptive Image Clustering

Input: Pairs of reference images: $\Lambda = \{\vec{G}, \vec{C}\}$; Error threshold: ε ; Minimal number of reference images required for training one DNN: μ ; Initial number of clusters on the top layer: N^0

Output: Trained DNN assemble Φ ; Hierarchy cluster assemble Ω .

```

1: Extract global descriptors of reference images  $\Lambda$ ;
2:  $l := 0$ ; // the top layer
3: while  $size(\Lambda) \geq \mu$  do
4:   Group  $\Lambda$  into  $N^l$  clusters  $\Omega_{(1...N^l, l)}$  on layer  $l$ ;
5:   Compute semantic histogram for each cluster  $\Omega_{(i, l)}$ ;
6:   Train a DNN  $\Phi_{(i, l)}$  for each cluster  $i$  on layer  $l$  using
     the reference images  $\Lambda_{(i, l)} = \{\vec{G}_{(i, l)}, \vec{C}_{(i, l)}\}$ 
7:   for each reference image  $I_{(i, l)}$  in  $\Lambda_{(i, l)}$  do
8:     Measure training error  $E(I)$ ;
9:     if  $E(I) \leq \varepsilon$  then
10:      Remove  $I$  from  $\Lambda_{(i, l)}$ ;
11:     end if
12:    $l := l + 1$ ,  $N^l := size(\Lambda_{(i, l)}) / \mu$ ;
13: end for
14: end while

```

1) *Semantic Histogram*: After scene-wise DNNs are trained, a straightforward colorization strategy is to find the nearest cluster for a target image and use the corresponding trained DNN to colorize it. However, it is very likely that the reference images in the searched cluster are globally similar but semantically different from the target images. For example, the nearest cluster for Figure 7(a) searched using only global image feature belongs to the “building” scene, which causes an unnatural colorization result, as shown in Figure 7 (b).

To address this problem, we incorporate the semantic histogram to search for the globally and semantically similar cluster. The number of bins is equal to the predefined object categories. And each bin that represents the percentage of pixels belongs to a certain object. In test phrase, we first search for the top- k nearest clusters by the Euclidean distance of global descriptors between the clusters and the target image, then find out the nearest cluster by the cosine similarity of semantic histogram within the initial k clusters. Figure 7 shows the performance could change significantly with and without semantic histogram.

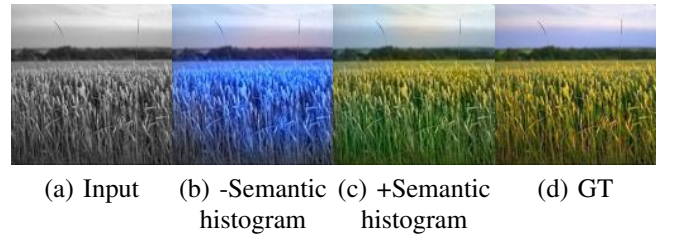


Fig. 7: Evaluation of semantic histogram. (a) is the input image. (b) is the colorization result when the semantic histogram is not used in nearest cluster searching. (c) is result after incorporating semantic histogram. (d) is the ground truth.

2) *The Evaluation of Image Clustering*: Figure 8 presents the PSNR distribution of 1519 test images with/without image clustering. Figure 9 shows the qualitative comparisons. It is seen that the proposed image clustering technique can improve the colorization accuracy and reduce the visible artifacts significantly, especially for the objects with large color variances (e.g. building, plant etc.)

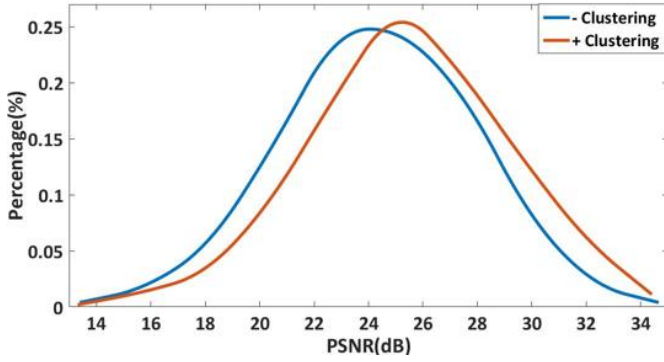


Fig. 8: The PSNR distribution with/without image clustering. It is seen that the proposed image clustering technique can improve the colorization accuracy.

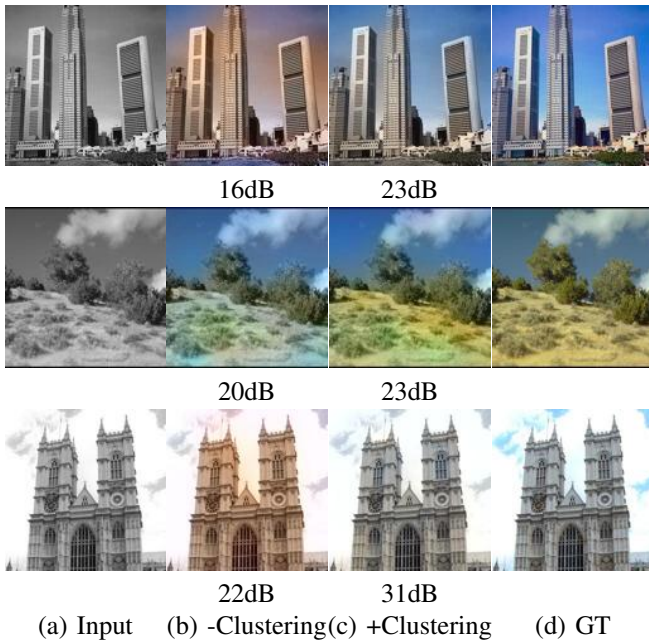


Fig. 9: Evaluation of the adaptive image clustering technique. (a) are the input images. (b) are the colorization results without image clustering. (c) are the results of the proposed method that utilizes image clustering. (d) are the ground truth of (a). The PSNR values computed from the colorization results and the ground truth are presented under the colorized images.

D. Difference from the State-of-the-art Colorization Methods

The previous algorithms [7], [8], [9], [10], [12] typically use one similar reference image or a set of similar reference images from which transfer color values to the target gray image. [9] is the state-of-art example-based method as it outperforms others in performance and application scope. However, its performance highly depends on given reference image as demonstrated in Figure 10. [9] can obtain a very good colorization result using a reference image containing identical object(s) as the target grayscale image. However, when the reference image is different from the target, its performance is quite low as shown in Figure 10 (h)-(i). To minimize the high dependence on a suitable reference image, our method

utilizes a large reference image database. It “finds” the most similar pixel from the database and “transfers” its color to the target pixel. This is why our approach is robust to different grayscale target images.



Fig. 10: The high dependence on a suitable reference image of Gupta et al. [9]. (a) is the input grayscale image. (b) is the color image obtained by the proposed method which is visually more accurate. (c) is the ground truth of (a). (d) is the first reference image for [9]. It has a similar scene as the (a). (e) is the second reference image that also has similar scene but lacks ‘beach’ object. (f) is the last reference image that is complete different from (a). The color images obtained from [9] w.r.t. the reference images in (d)-(f) are presented in (g)-(i), respectively. The PSNR values computed from the colorization results and the ground truth are presented under the colorized images.

Intuitively, one reference image cannot comprise all suitable correspondences for pixels in the target grayscale image. This is why the performance of [9] highly depends on a suitable reference image. As shown in Figure 11, using a couple of similar reference images could improve their colorization result. However, when the reference images contain multiple objects (e.g. door, window, building etc.), their colorization result becomes unnatural, although some of the reference images are similar to the target. This is due to the significant amount of noise residing in feature matching (between the reference images and the target image). For instance, we noticed that the lake in Figure 10(a) was matched to the door in Figure 11(e)), and the sky was matched to the building in Figure 11(f).

Experiments demonstrate that deep learning techniques are well-suited for a large reference image database. The deep neural network helps to combine the various features of a pixel and computes the corresponding chrominance values. Additionally,

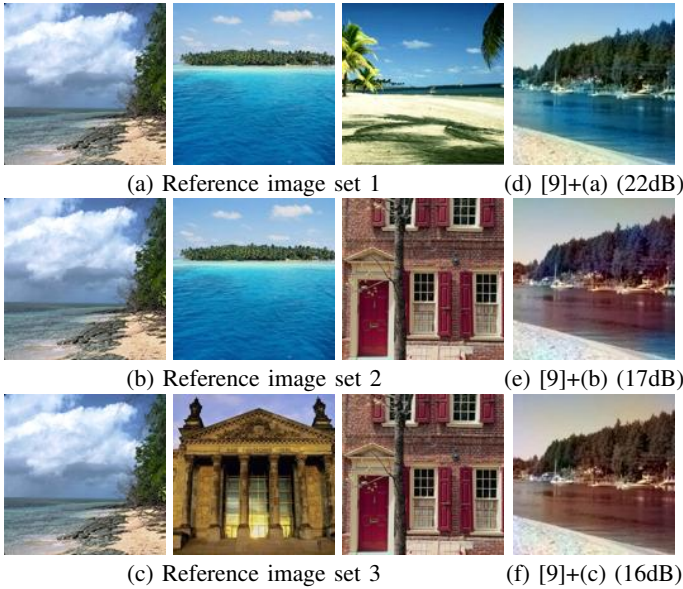


Fig. 11: Gupta et al.[9] with multiple reference images. The target grayscale image is the same as Figure 10(a). (a)-(c) are different reference images and (d)-(f) are the corresponding colorization results. Note that the best performance can be achieved when sufficient similar reference images are used.

the state-of-the-art methods are very slow because they have to find the most similar pixels (or super-pixels) from massive candidates. In comparison, the deep neural network is tailored to massive data. Although the training of neural networks is slow especially when the database is large, colorizing a 256×256 grayscale image using the trained neural network assemble takes only 6.780 seconds in Matlab.

More recently, Deshpande et al. [20] propose an automatic colorization framework. Similar to our method, [20] solves this problem by minimizing a quadratic objective function, and also proposes an post-processing technique to improve their colorization performance. The main differences lie in the following aspects:

- 1) The proposed deep neural networks learn the mapping function automatically, so that we need not design the objective function carefully by hand or search for massive hyper-parameters like [20];
- 2) To achieve good performances, [20] requires a suitable scene histogram in their refinement step. Their best colorization results are typically obtained by using the ground-truth scene histogram. By contrast, no such spatial prior is required for the proposed method.
- 3) The proposed model colorizes a target image at a much higher speed than [20]. It takes only 6.780 seconds to colorize a 256×256 using the proposed model while [20] requires 251.709 seconds and more time to adjust the histograms in their refinement step.

IV. EXPERIMENTAL RESULTS

The proposed colorization neural network assemble is trained on 2344 images from the SIFT Flow database (a subset of SUN Attribution database [32]). We evaluate the proposed

TABLE I: Performance comparison of semantic segmentation model. The first column lists the existing state-of-art scene parsing algorithms, and the second column shows the version of training and test images. The last column presents the standard metric (i.e. pixel accuracy) for evaluation.

Methods	Image Version	Pixel Acc.
Long et al. [25]	color	85.2
Long et al. [25]	grayscale	78.9
Liu et al. [34]	color	76.7
Tighe et al.[35] 1	color	75.6
Tighe et al.[35] 2	color	78.6
Farabet et al. [36] 1	color	72.3
Farabet et al. [36] 2	color	78.5
Pinheiro et al. [37]	color	77.7

model on 1519 images from Sun database [33]. Each image is segmented into a number of object regions and a total of 33 object categories¹ are used (e.g. building, car, sea etc.). The neural network has an input layer, three hidden layers and one output layer. According to our experiments, using more hidden layers cannot further improve the colorization results. A 49-dimension (7×7) patch feature, a 32-dimension DAISY feature [24] (4 locations and 8 orientations) and a 33-dimension semantic feature are extracted at each pixel location. Thus, there are a total of 114 neurons in the input layer. This paper perceptually sets the number of neurons in the hidden layer to half of that in the input layer and 2 neurons in the output layer (which correspond to the chrominance values). The parameters ε , μ , N^0 for the proposed adaptive image clustering are set to -26dB, 80 and 24 respectively. We use gist feature [29] as the global image descriptor in our experiment.

A. Scene Parsing on Grayscale Image

We retrained the semantic segmentation model proposed by [25] using the grayscale version of images from SIFT Flow dataset and evaluated the trained model on the standard 200 test images. As shown in Table I, [25] outperforms other algorithms [34], [35], [36], [37] in terms of pixel accuracy, whether trained by color or grayscale images. It also proves that the color information is useful for scene parsing, as the best performance is achieved by training [25] using color images. We verify that the retrained model of [25] on grayscale images is sufficient enough for our colorization work.

B. Comparisons with State-of-the-Arts

Figure 6 compares our colorization results with the state-of-the-art colorization methods [7], [9], [10], [12]. The performance of these colorization methods is very high when an “optimal” reference image is used (e.g., containing the same objects as the target grayscale image), as shown in [7], [9], [10], [12]. However, the performance may drop significantly when the reference image is only similar to the target grayscale

¹There is one error in [1]. Only 33 (instead of 47) object categories were used in [1].

image. The proposed method does not have this limitation due to the use of a large reference image database as shown in Figure 1 (a).

Figure 12 shows the comparison with [20]. It is seen that [20] performs well when a suitable scene histogram is used in their refinement step, but visible artifacts still appear frequently. By contrast, the proposed method generates more natural colorization results with higher spatial coherency and fewer artifacts, and no spatial priors are required.

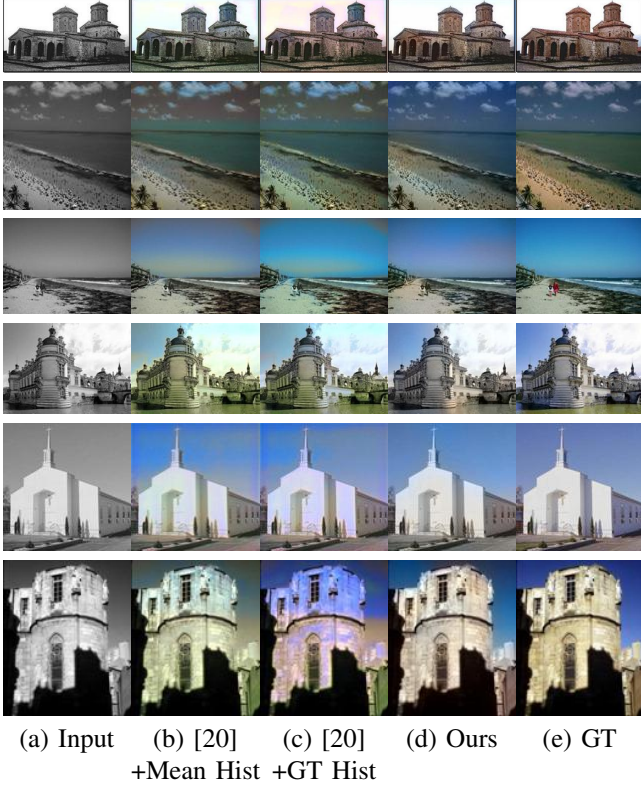


Fig. 12: Comparison with Deshpande et al. [20]. (a) are the input grayscale images. (b) are the results generated by [20] using mean color histogram computed from reference images. (c) are the results of [20] using ground-truth color histogram of (a). (d) are the results of the proposed model. (e) are the ground truth of (a).

C. Colorization in Different Global Styles

One problem of [1] is that it colorizes the target grayscale image in one global style. For example, as shown in Figure 13 (b), all grayscale images are colorized in a daytime style automatically. Although these colorization results are visually reasonable, it is possible that the user has special requirements on the colorization style (e.g. dusk). However, given a grayscale image, it is very challenging to recognize whether it belongs to daytime or dusk even by human eyes, which makes it hard to generate more than one colorization styles using an uniform neural network. An alternative is to train a specific neural network for the required global style. Our experiments show that the proposed model is flexible to learn different global styles, as shown in Figure 13.

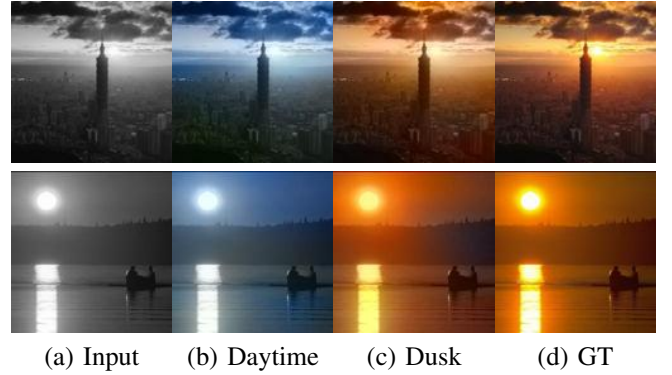


Fig. 13: Colorization in daytime/dusk style. (a) is the input image. (b) is the colorization results in daytime style. (c) is results in dusk style. (d) is the ground truth.

TABLE II: Running Time (seconds) on images of different resolutions, and comparison to Deshpande et al. [20]. Note that we only compare with [20] here, since both the proposed method and [20] are fully-automatic while other colorization methods [2], [3], [4], [5], [6], [7], [8], [9], [10], [11], [12] typically require efforts from the user, which makes it hard to measure their running time.

Image Size	256×256	512×512	1024×1024
Proposed	6.780	17.413	63.142
[20]	251.709	712.149	5789.075

D. Running Time

The proposed model is able to process images of any resolutions at a high speed. Table II shows the average running time on images of different resolutions on a computer equipped with Intel® Xeon® @ 2.30GHz CPU, along with the comparison to [20]². It is seen that the proposed model is much faster than [20], and our running time increases nearly linearly with the image resolution.

E. More Colorization Results

Figure 14 presents more colorization results obtained from the proposed method with respect to the ground-truth color images³. Figure 14 demonstrates that there are almost not visible artifacts in the color images generated using the proposed method, and these images are visually very similar to the ground truth.

V. LIMITATIONS

The proposed colorization is fully-automatic and thus is normally more robust than the traditional methods. However, it relies on machine learning techniques and has its own limitations. For instance, it is supposed to be trained on a huge reference image database which contains all possible objects. This is impossible in practice. For instance, the current model was trained on real images and thus is invalid for the synthetic

²We use the source code released by [20] in our experiment. <http://vision.cs.illinois.edu/projects/lscolor>

³More colorization results are available at authors' website. <http://www.cs.cityu.edu.hk/~qiyang/publications/iccv15/>

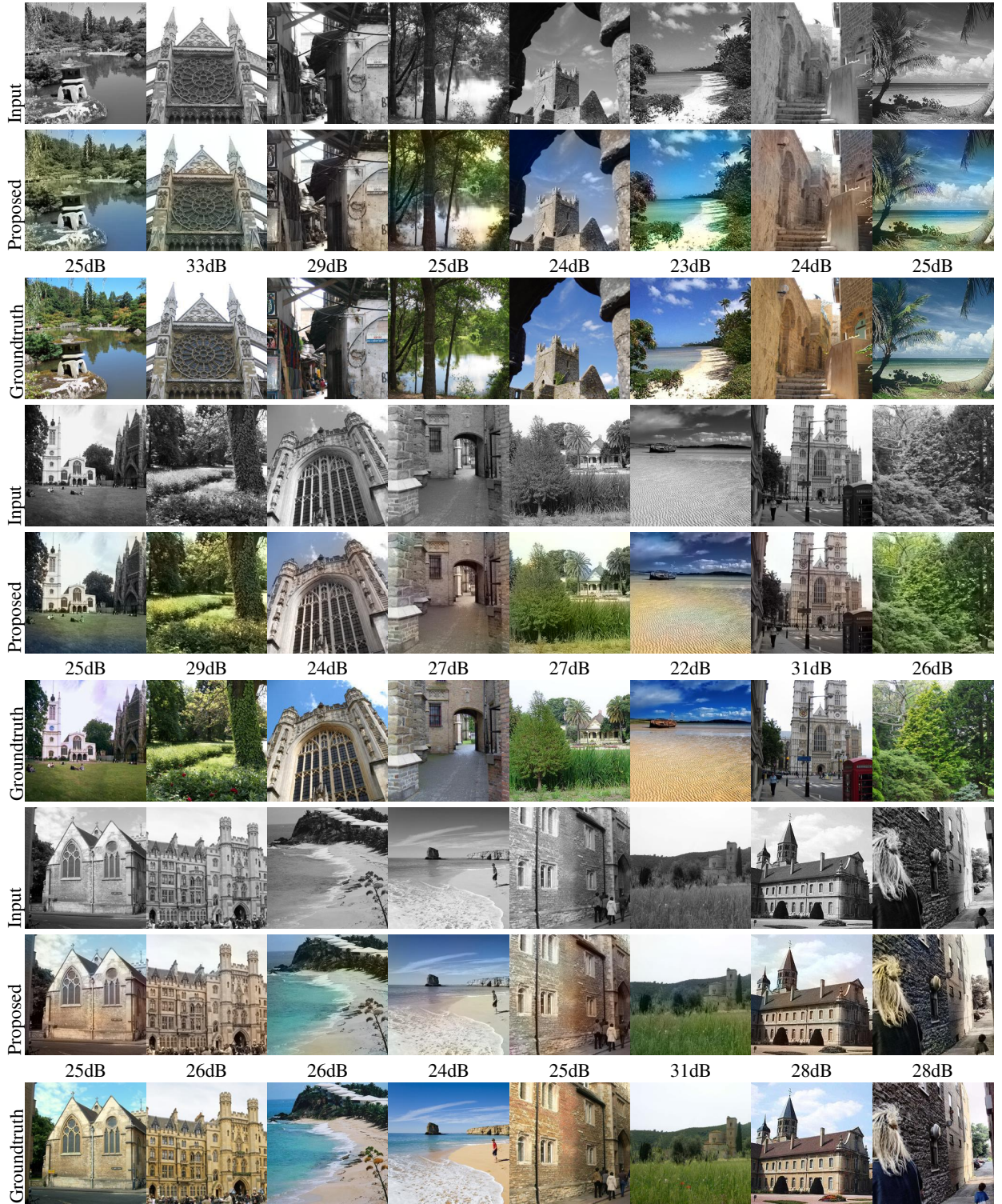


Fig. 14: Comparison with the ground truth. The 1st/4th/7th rows present the input grayscale images from different categories. Colorization results obtained from the proposed method are presented in the 2nd/5th/8th rows. The 3rd/6th/9th row presents the corresponding ground-truth color images, and the PSNR values computed from the colorization results and the ground truth are presented under the colorized images.

image. It is also impossible to recover the color information lost due to color to grayscale transformation. Nevertheless, this is a limitation to all state-of-the-art colorization method. Two failure cases are presented in Figure 15.

VI. CONCLUDING REMARKS

This paper presents a novel, fully-automatic colorization method using deep neural networks to minimize user effort and the dependence on the example color images. Informative yet discriminative features including patch feature, DAISY

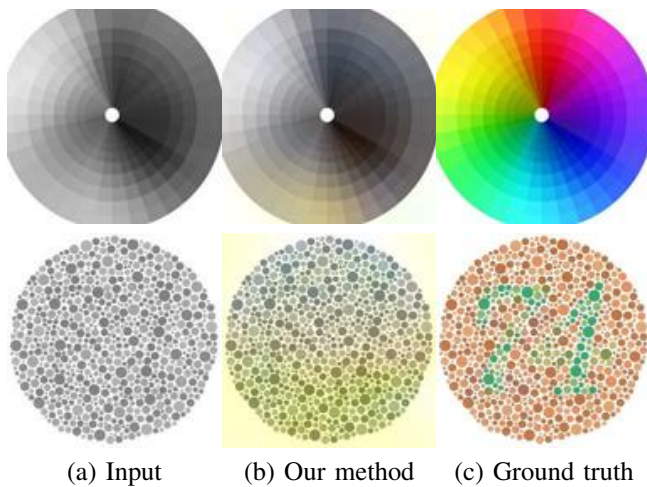


Fig. 15: Limitations of our method. Our method is not suitable for synthetic images and cannot recover the information lost during color to grayscale conversion. Note that the green number in the last row of (c) disappears in the corresponding grayscale image in (a).

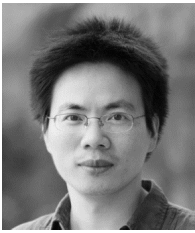
feature and a new semantic feature are extracted and serve as the input to the neural network. An adaptive image clustering technique is proposed to incorporate the global image information. The output chrominance values are further refined using joint bilateral filtering to ensure artifact-free colorization quality. Numerous experiments demonstrate that our method outperforms the state-of-art algorithms both in terms of quality and speed.

REFERENCES

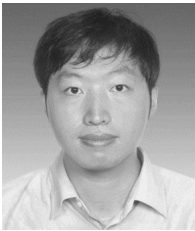
- [1] Z. Cheng, Q. Yang, and B. Sheng, "Deep colorization," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 415–423.
- [2] Y.-C. Huang, Y.-S. Tung, J.-C. Chen, S.-W. Wang, and J.-L. Wu, "An adaptive edge detection based colorization algorithm and its applications," in *Proceedings of the 13th Annual ACM International Conference on Multimedia*, ser. MULTIMEDIA '05, 2005, pp. 351–354.
- [3] A. Levin, D. Lischinski, and Y. Weiss, "Colorization using optimization," in *ACM SIGGRAPH 2004 Papers*, 2004, pp. 689–694.
- [4] Q. Luan, F. Wen, D. Cohen-Or, L. Liang, Y.-Q. Xu, and H.-Y. Shum, "Natural image colorization," in *Proceedings of the 18th Eurographics Conference on Rendering Techniques*, ser. EGSR'07, 2007, pp. 309–320.
- [5] Y. Qu, T.-T. Wong, and P.-A. Heng, "Manga colorization," in *ACM SIGGRAPH 2006 Papers*, ser. SIGGRAPH '06, 2006, pp. 1214–1220.
- [6] L. Yatziv and G. Sapiro, "Fast image and video colorization using chrominance blending," *Trans. Img. Proc.*, vol. 15, no. 5, pp. 1120–1129, 2006.
- [7] G. Charpiat, M. Hofmann, and B. Schölkopf, "Automatic image colorization via multimodal predictions," in *ECCV*. Springer, 2008, pp. 126–139.
- [8] A. Y.-S. Chia, S. Zhuo, R. K. Gupta, Y.-W. Tai, S.-Y. Cho, P. Tan, and S. Lin, "Semantic colorization with internet images," in *TOG*, vol. 30, no. 6. ACM, 2011, p. 156.
- [9] R. K. Gupta, A. Y.-S. Chia, D. Rajan, E. S. Ng, and H. Zhiyong, "Image colorization using similar images," in *ACM international conference on Multimedia*. ACM, 2012, pp. 369–378.
- [10] R. Irony, D. Cohen-Or, and D. Lischinski, "Colorization by example," in *Eurographics Symp. on Rendering*, vol. 2. Citeseer, 2005.
- [11] X. Liu, L. Wan, Y. Qu, T.-T. Wong, S. Lin, C.-S. Leung, and P.-A. Heng, "Intrinsic colorization," in *TOG*, vol. 27, no. 5. ACM, 2008, p. 152.
- [12] T. Welsh, M. Ashikhmin, and K. Mueller, "Transferring color to greyscale images," *ACM Trans. Graph.*, vol. 21, no. 3, pp. 277–280, Jul. 2002.
- [13] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 1026–1034.
- [14] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.
- [15] W. Ouyang and X. Wang, "Joint deep learning for pedestrian detection," in *ICCV*. IEEE, 2013, pp. 2056–2063.
- [16] X. Zeng, W. Ouyang, and X. Wang, "Multi-stage contextual deep learning for pedestrian detection," in *ICCV*. IEEE, 2013, pp. 121–128.
- [17] C. Dong, C. C. Loy, K. He, and X. Tang, "Learning a deep convolutional network for image super-resolution," in *ECCV*. Springer, 2014, pp. 184–199.
- [18] Z. Yan, H. Zhang, B. Wang, S. Paris, and Y. Yu, "Automatic photo adjustment using deep neural networks," *ACM Trans. Graph.*, vol. 35, no. 2, pp. 11:1–11:15, Feb. 2016.
- [19] S. Zheng, A. Yuille, and Z. Tu, "Detecting object boundaries using low-, mid-, and high-level information," *CVIU*, vol. 114, no. 10, pp. 1055–1067, 2010.
- [20] A. Deshpande, J. Rock, and D. Forsyth, "Learning large-scale automatic image colorization," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 567–575.
- [21] A. Hertzmann, C. E. Jacobs, N. Oliver, B. Curless, and D. H. Salesin, "Image analogies," in *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques*, ser. SIGGRAPH '01, 2001, pp. 327–340.
- [22] E. Reinhard, M. Ashikhmin, B. Gooch, and P. Shirley, "Color transfer between images," *IEEE Comput. Graph. Appl.*, vol. 21, no. 5, pp. 34–41, 2001.
- [23] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural networks*, vol. 2, no. 5, pp. 359–366, 1989.
- [24] E. Tola, V. Lepetit, and P. Fua, "A fast local descriptor for dense matching," in *CVPR*. IEEE, 2008, pp. 1–8.
- [25] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 3431–3440.
- [26] E. S. Gastal and M. M. Oliveira, "Domain transform for edge-aware image and video processing," in *TOG*, vol. 30, no. 4. ACM, 2011, p. 69.
- [27] G. Petschnigg, M. Agrawala, H. Hoppe, R. Szeliski, M. Cohen, and K. Toyama, "Digital photography with flash and no-flash image pairs," *ToG*, 2004.
- [28] D. Comaniciu and P. Meer, "Mean shift: A robust approach toward feature space analysis," *PAMI*, vol. 24, no. 5, pp. 603–619, 2002.
- [29] C. Siagian and L. Itti, "Rapid biologically-inspired scene classification using features shared with visual attention," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 29, no. 2, pp. 300–312, 2007.
- [30] P. Ren, Y. Dong, S. Lin, X. Tong, and B. Guo, "Image based relighting using neural networks," *ACM Transactions on Graphics (TOG)*, vol. 34, no. 4, p. 111, 2015.
- [31] M. J. Turmon and T. L. Fine, "Sample size requirements for feedforward neural networks," *Advances in Neural Information Processing Systems*, pp. 327–334, 1995.
- [32] G. Patterson and J. Hays, "Sun attribute database: Discovering, annotating, and recognizing scene attributes," in *CVPR*. IEEE, 2012, pp. 2751–2758.
- [33] J. Xiao, J. Hays, K. A. Ehinger, A. Oliva, and A. Torralba, "Sun database: Large-scale scene recognition from abbey to zoo," in *Computer vision and pattern recognition (CVPR), 2010 IEEE conference on*. IEEE, 2010, pp. 3485–3492.
- [34] C. Liu, J. Yuen, and A. Torralba, "Sift flow: Dense correspondence across scenes and its applications," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 33, no. 5, pp. 978–994, 2011.
- [35] J. Tighe and S. Lazebnik, "Finding things: Image parsing with regions and per-exemplar detectors," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2013, pp. 3001–3008.
- [36] C. Farabet, C. Couprie, L. Najman, and Y. LeCun, "Learning hierarchical features for scene labeling," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 35, no. 8, pp. 1915–1929, 2013.
- [37] P. H. Pinheiro and R. Collobert, "Recurrent convolutional neural networks for scene parsing," in *Proceedings of The International Conference on Machine Learning*, 2014, pp. 82–90.



Zezhou Cheng received the B.Eng. degree in computer science and technology from Sichuan University, Chengdu, China. He is currently pursuing the Ph.D. degree in the Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai, China. His research interests include image processing, computer vision and machine learning.



Qingxiong Yang is an Assistant Professor in the Department of Computer Science at City University of Hong Kong. He obtained his B.Eng. degree in Electronic Engineering & Information Science from University of Science & Technology of China (USTC) in 2004 and PhD degree in Electrical & Computer Engineering from University of Illinois at Urbana-Champaign in 2010. His research interests reside in Computer Vision and Computer Graphics. He won the best student paper award at MMSP 2010 and best demo at CVPR 2007.



Bin Sheng received the B.A. degree in english and B.E. degree in computer science from Huazhong University of Science and Technology, Wuhan, China, in 2004, the M.S. degree in software engineering from the University of Macau, Macau, China, in 2007, and the Ph.D. degree in computer science from The Chinese University of Hong Kong, Hong Kong, in 2011. He is currently an Associate Professor with the Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai, China. His research interests include virtual reality, computer graphics, and image-based techniques.