

Deep Colorization

Zezhou Cheng

Shanghai Jiao Tong University
chengzezhou@sjtu.edu.cn

Qingxiong Yang

City University of Hong Kong
qiyang@cityu.edu.hk

Bin Sheng*

Shanghai Jiao Tong University
shengbin@sjtu.edu.cn

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 reformulates 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. Numerous experiments demonstrate that our method outperforms the state-of-art algorithms both in terms of quality and speed.

1. 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 [10, 13, 16, 20, 25] and example-based colorization [1, 2, 6, 11, 14, 23]. 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, [23] proposes an example-based method which is later further improved by [1, 11]. 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. [2, 14] 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. [14] 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). [2] 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, [1, 2, 11, 23] 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 to some extent (e.g. [7]) and deep learning techniques have been demonstrated to be very effective on various computer vision and image processing applications including image classification [12], pedestrian detection [17, 26], image super-resolution [4], photo adjustment [24] 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, animal, building, sea, mountain etc.) is used for training the neural networks. Some example reference images are presented in Figure 1 (b). Although the training is very 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 coloriza-

*Correspondence author.

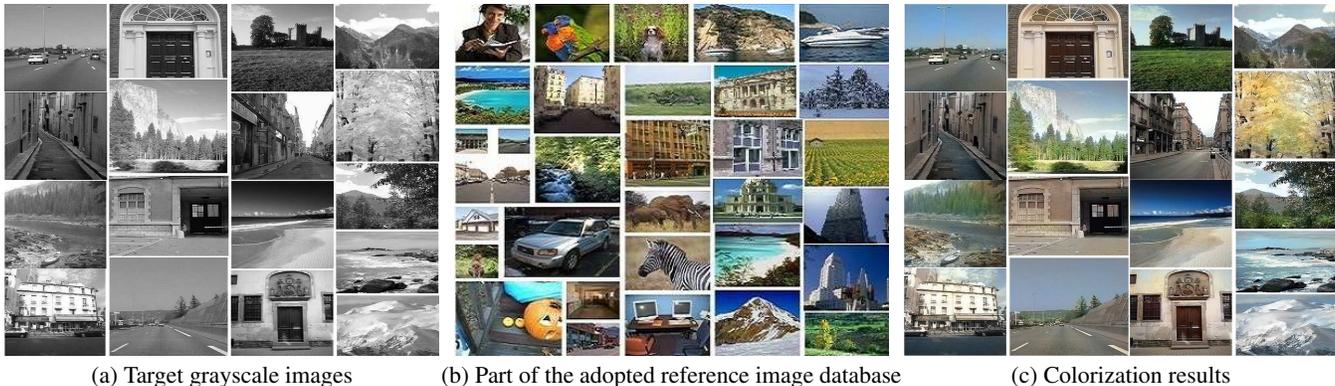


Figure 1. The colorization results of our full-automatic method. Our system utilizes a large database of colorful reference images as shown in (b). After the training of neural networks, the learned model is used directly to colorize the target gray scale images in (a). The colorization results are presented in (c).

tion 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 very useful for low-level vision problems (*e.g.* image enhancement [24], edge detection [27]). Because image colorization is typically semantic-aware, we propose a new semantic feature descriptor to incorporate the semantic-awareness into our colorization model.

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

2. Related Work

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

Scribble-based colorization Levin et al. [13] 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. [10] develop an adaptive edge detection algorithm to reduce the color bleeding artifact around the region boundaries. Yatziv et al. [25] colorize the pixels using a weighted combination of user scribbles. Qu et al. [20] and Luan et

al. [16] 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 separated 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 [8] and the color transfer technology [21], Welsh et al. [23] 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 [11] by taking into account the texture feature. Charpiat et al. [1] propose a global optimization algorithm to colorize a pixel. Gupta et al. [6] 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) and 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.[14] and Chia et al. [2] utilize the massive image data on the Internet. Liu et al.[14] 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. [2] propose an image fil-

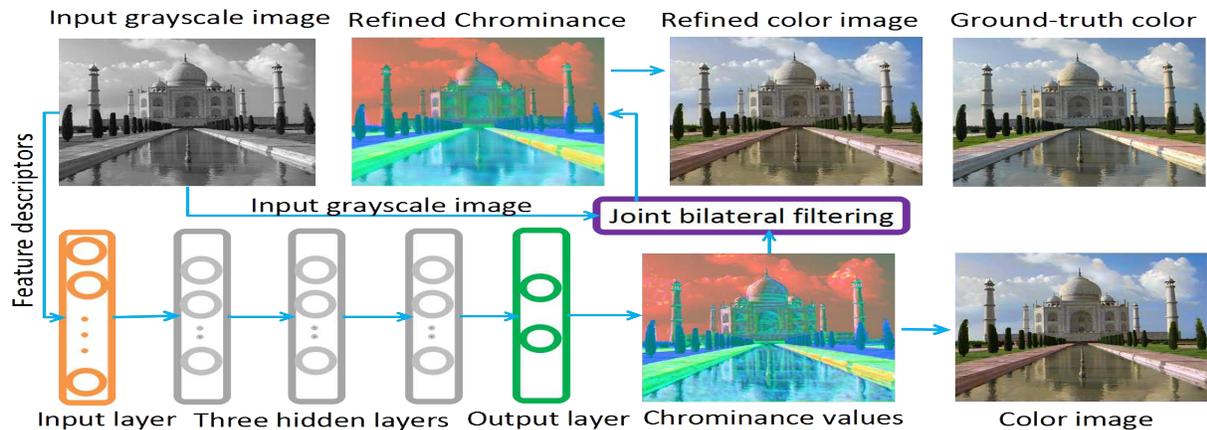


Figure 2. Overview of the proposed colorization method and the architecture of the adopted deep neural network. The feature descriptors will be extracted at each pixel and serve as the input of the neural network. Each connection between pairs of neurons is associated with a weight to be learned from 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).

ter 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.

In contrast to the previous colorization methods, the proposed method is fully automatic by utilizing a large set of reference images from different categories (*e.g.*, animal, outdoor, indoor) with various objects (*e.g.*, tree, person, panda, car etc.).

3. Our Metric

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

Algorithm 1 Image Colorization – Training Step

Input: Pairs of reference images: $\Lambda = \{\vec{G}, \vec{C}\}$.
Output: A trained neural network.

1. Compute feature descriptors \vec{x} at sampled pixels in \vec{G} and the corresponding chrominance values \vec{y} in \vec{C} ;
 2. Construct a deep neural network;
 3. Train the deep neural network 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.
Output: A corresponding color image: \hat{I} .

1. Extract a feature descriptor at each pixel location in I ;
 2. Send feature descriptors extracted from I to the trained neural network to obtain the corresponding chrominance values;
 3. Refine the chrominance values to remove potential artifacts;
 4. Combine the refined chrominance values and I to obtain the color image \hat{I} .
-

3.1. 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.

3.1.1 Formulation

A deep neural network is a universal approximator that can represent arbitrarily complex continuous functions [9]. Given a set of exemplars $\Lambda = \{\vec{G}, \vec{C}\}$, where \vec{G} are grayscale images and \vec{C} are corresponding color images 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 we can use \mathcal{F} to convert a new gray image to color image. In our model, we employ the YUV color space, 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. 3.2. 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 :

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

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

3.1.2 Architecture

Deep neural networks (DNNs) typically consist of one input layer, multiple hidden layers and one output layer. Each layer can comprise various number 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 $(l-1)^{\text{th}}$ 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[12]). 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[12] 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 2.

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.

3.2. Feature descriptor

Feature design is key to the success of the proposed colorization method. There are massive candidate image fea-

tures 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 the 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.

3.2.1 Low-level patch feature

Intuitively, there exists 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 denote the sequential grayscale values in a 7×7 patch center at p , x_p 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 3 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 3(b)).



(a)Input (b)-patch feature (c)+patch feature

Figure 3. Evaluation of patch feature. (a) is the target grayscale image. (b) removes the low-level patch feature and (c) includes all the proposed features.

3.2.2 Mid-level DAISY feature

DAISY is a fast local descriptor for dense matching [22]. 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 a pixel location p in a grayscale image and is denote as x_p^M . Figure 4 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 4(c). A post-processing step will be introduced in Section 3.2.5 to reduce the artifacts and result is presented in Figure 4(d). Furthermore, we can see that our result is comparable to Liu et al. [14] (which requires a rigid-shape target object and identical reference objects) and Chia et al. [2] (which requires manual segmentation and identification of the foreground objects), although our method is fully-automatic.

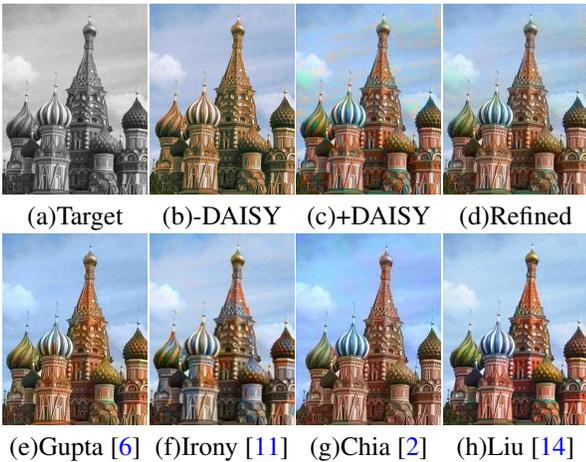


Figure 4. Evaluation of DAISY feature. (a) is the target gray scale image. (b) is our result without DAISY feature. (c) is our result after incorporating DAISY feature into our model. (d) is the final result after artifact removal (see Sec. 3.2.5 for details). (e)-(h) presents results obtained with the state-of-the-art colorizations. Although the proposed method is fully-automatic, its performance is comparable to the state-of-the-art.

3.2.3 High-level Semantic feature

Patch and DAISY feature 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 a image have demonstrated its importance and virtues in some fields (*e.g.* image enhancement [24], edge detection [27]). Consider 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 utilize the state-of-art scene parsing algorithm [15] 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 [5] 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^H .

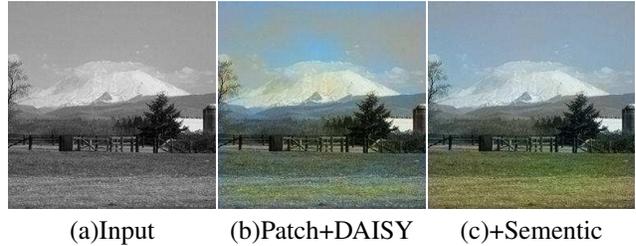


Figure 5. Importance of semantic feature. (a) is the target grayscale image. (b) is the colorization result using patch and DAISY features only. (c) is the result using patch, DAISY and semantic features.

Figure 5 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 5.

3.2.4 Global features

Unlike the previous example-based methods, we don't incorporate the global image features (*e.g.*, gist, histogram etc.) into our model. State-of-the-art methods [2], [14] typically utilize the global features to construct an image filter, and then use it to select similar reference images from a large image set automatically. However, it is very likely that the image filter selects a reference image that is globally similar but semantically different from the target image. When we incorporate the global features into our model, it would produce an unnatural colorization result as shown in Figure 6.

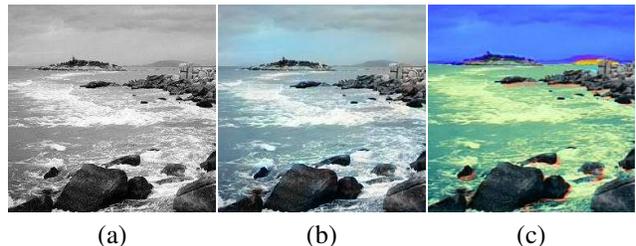


Figure 6. Evaluation of global features. (a) is the target gray scale image. (b) is our final result using patch, DAISY and semantic features. (c) is the colorization result when we incorporate the global features into our model.

3.2.5 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 [19]. 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 7 presents the result before and after chrominance refinement. Note that most of the visible artifacts can be successfully removed.

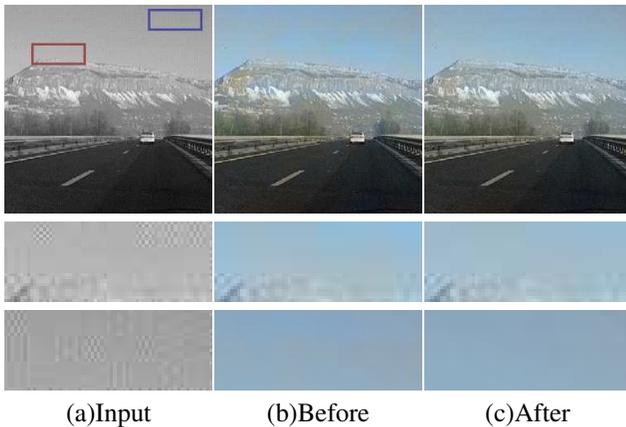


Figure 7. Chrominance refinement using joint bilateral filtering [19]. From (a) to (c): target grayscale image, colorization results before and after chrominance refinement, respectively. Note that the artifacts in (b) are successfully removed from (c).

3.3. Difference from the state-of-the-art colorization methods

The previous algorithms [1, 2, 6, 11, 23] typically use one similar reference image or a set of similar reference images from which transfer color values to the target gray image. [6] is the state-of-the-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. [6] 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.

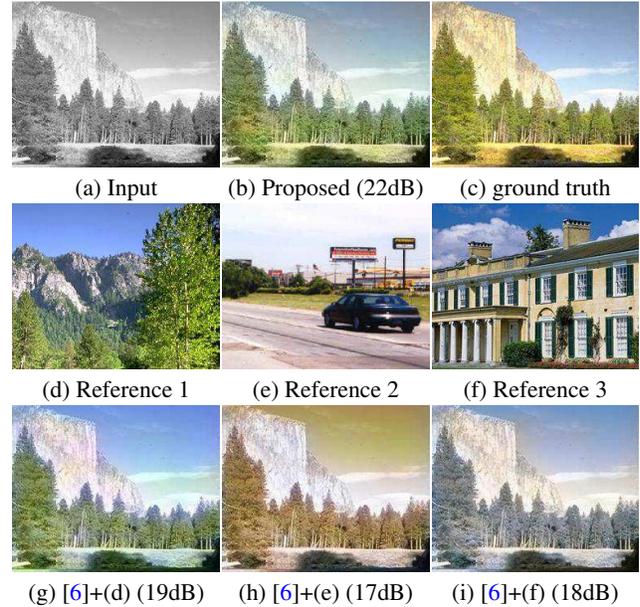


Figure 10. The high dependence on a suitable reference image of Gupta et al. [6]. (a) is the input grayscale image. (b) is the color image obtained from the proposed method which is visually more accurate. (c) is the ground truth of (a). (d) is the first reference image for [6]. It has a similar scene as the (a). (e) is the second reference image. It is semantically different from (a). (f) is the last reference image that is complete different from (a). The color images obtained from [6] 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 [6] 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 is 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 grassland in Figure 10(a) was matched to the wall in Figure 11(e), and the sky was matched to the building in Figure 11(f).

Experiments demonstrate that deep learning technique

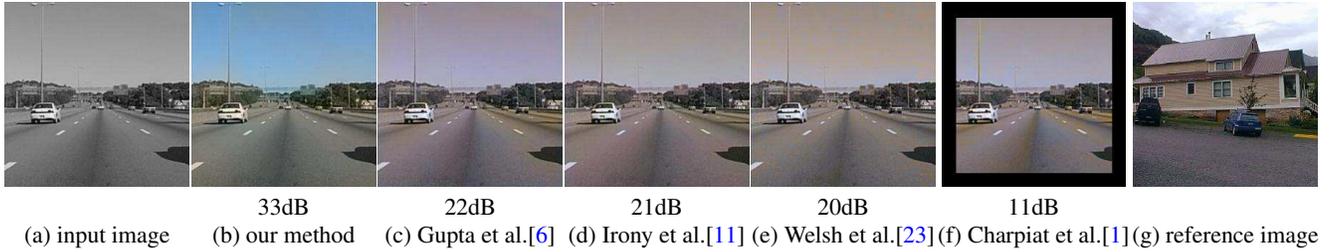


Figure 8. Comparison with the state-of-art colorization methods [1, 6, 11, 23]. (c)-(f) use (g) as the reference image, while the proposed method adopts a large reference image dataset. The reference image contains similar objects as the target grayscale image (e.g., road, trees, building, cars). It 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 [11] are computed by mean shift algorithm [3]. The PSNR values computed from the colorization results and ground truth are presented under the colorized images.

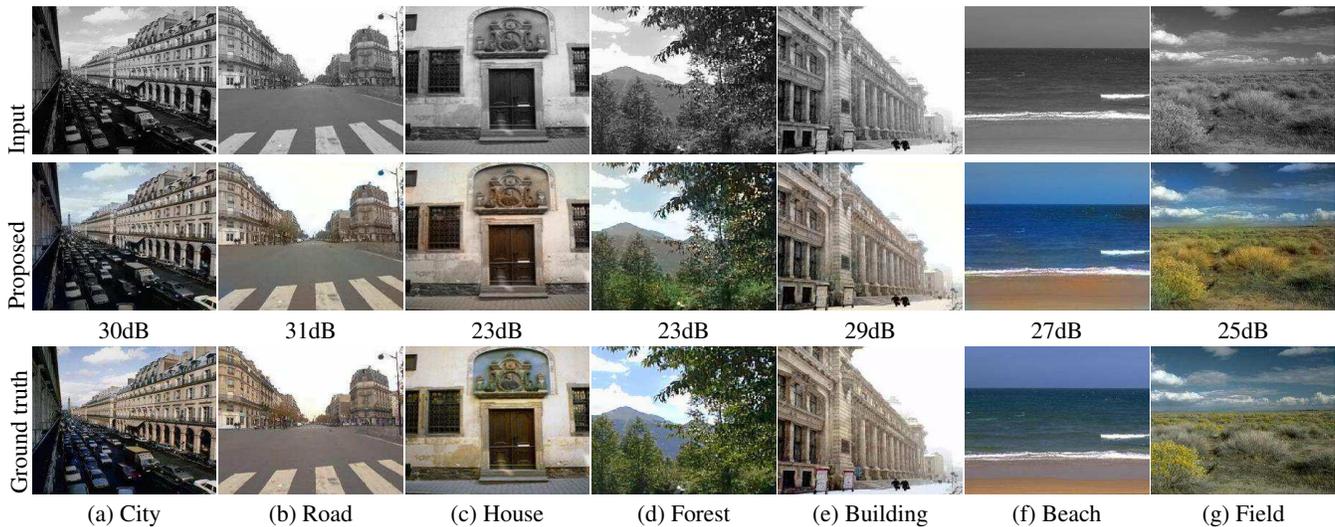


Figure 9. Comparison with the ground truth. The first row presents the input grayscale images from different categories. Colorization results obtained from the proposed method are presented in the second row. The last row presents the corresponding ground-truth color images, and the PSNR values computed from the colorization results and the ground truth are presented under the colorized images.

is 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, 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 network is slow especially when the database is large, colorizing a 256×256 grayscale image using the trained neural network takes only 4.9 seconds in Matlab.

4. Experimental Results

The proposed colorization neural network is trained on 2688 images from the Sun database [18]. Each image is segmented into a number of object regions and a total of 47 object categories are used (e.g. building, car, sea etc.). The neural network has an input layer, three hidden lay-

ers 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 [22] (4 locations and 8 orientations) and a 47-dimension semantic feature are extracted at each pixel location. Thus, there are a total of 128 neurons in the input layer. This paper perceptually set 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).

Figure 8 compares our colorization results with the state-of-the-art colorization methods [1, 6, 11, 23]. 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 [1, 6, 11, 23]. However, the performance may drop significantly when the reference image is only similar to the target grayscale image. The proposed method does not have this limitation due to the use of a large reference image database

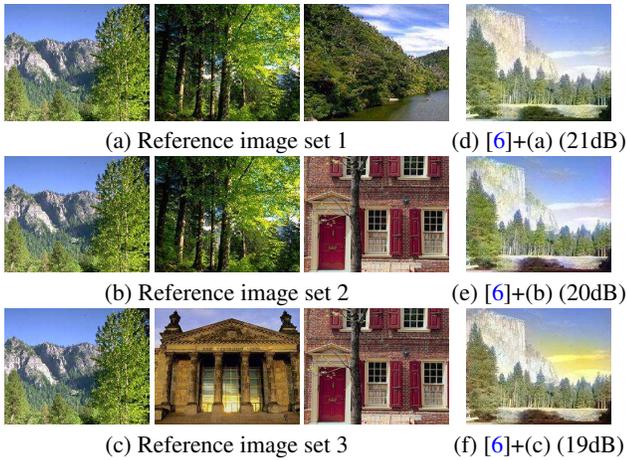


Figure 11. Gupta et al.[6] with multiple reference images. The target grayscale image is the same as Figure 10(a). (a)-(c) are different reference images and (d)-(f) are the corresponding colorization results. Note that the best performance can be achieved when sufficient similar reference images are used.

as shown in Figure 1(b).

To test the influence of sample size on our model, we utilized different sizes of samples to train our DNN, and then colorized 1344 grayscale images. Let Ψ denote our training set. The sample size is denoted as $\delta = \text{size}(\Psi)$. As shown in Figure 12, we can see that more samples will result in more accurate colorization results.

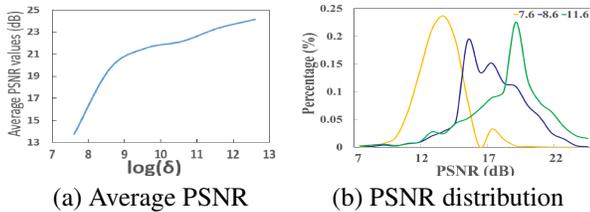


Figure 12. Influence of sample size on the colorization results. (a) and (b) presents the average and the distribution of PSNR using different sample size. The yellow, blue and green curves in (b) represent the PSNR distribution when $\log(\delta)$ are assigned with 7.6, 8.6 and 11.6 respectively.

Figure 9 presents more colorization results obtained from the proposed method with respect to the ground-truth color images. Figure 9 demonstrates that there is almost not visible artifact in the color images generated using the proposed method, and these images are visually very similar to the ground truth.

5. 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 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 13.

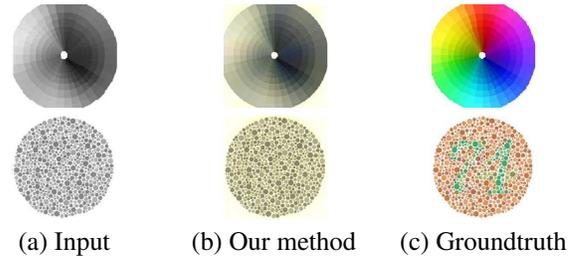


Figure 13. Limitations of our method. Our method is not suitable for synthetic images and cannot recover the information lost during color to grayscale conversion. Note that the green number in the last row of (c) disappears in the corresponding grayscale image in (a).

6. Concluding Remarks

This paper presents a novel, full-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 feature and a new semantic feature are extracted and serve as the input to the neural network. 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.

Acknowledgements: The work is supported by the National Natural Science Foundation of China (Nos. 61202154, 61572316, 61370174 and 61133009), the National Basic Research Project of China (No. 2011CB302203), National High-tech R&D Program of China (863 Program)(Grant No. 2015AA011604), National Key Technology R&D Program (No.2012BAH55F02). This work is also supported by the Early Career Scheme through the Research Grants Council, University Grants Committee, Hong Kong, under Grant CityU 21201914, and Shanghai Pujiang Program (No. 13PJ1404500), the Interdisciplinary Program of Shanghai Jiao Tong University (No.14JCY10), and the Open Project Program of the State Key Lab of CAD&CG (No. A1401), Zhejiang University. We gratefully acknowledge the support of NVIDIA Corporation with the donation of the Tesla K40 GPU used for this research.

References

- [1] G. Charpiat, M. Hofmann, and B. Schölkopf. Automatic image colorization via multimodal predictions. In *ECCV*, pages 126–139. Springer, 2008. 1, 2, 6, 7
- [2] 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*, volume 30, page 156. ACM, 2011. 1, 2, 5, 6
- [3] D. Comaniciu and P. Meer. Mean shift: A robust approach toward feature space analysis. *PAMI*, 24(5):603–619, 2002. 7
- [4] C. Dong, C. C. Loy, K. He, and X. Tang. Learning a deep convolutional network for image super-resolution. In *ECCV*, pages 184–199. Springer, 2014. 1
- [5] E. S. Gastal and M. M. Oliveira. Domain transform for edge-aware image and video processing. In *TOG*, volume 30, page 69. ACM, 2011. 5
- [6] 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*, pages 369–378. ACM, 2012. 1, 2, 5, 6, 7, 8
- [7] K. He, X. Zhang, S. Ren, and J. Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. *arXiv preprint arXiv:1502.01852*, 2015. 1
- [8] 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*, SIGGRAPH '01, pages 327–340, 2001. 2
- [9] K. Hornik, M. Stinchcombe, and H. White. Multilayer feedforward networks are universal approximators. *Neural networks*, 2(5):359–366, 1989. 3
- [10] 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*, MULTIMEDIA '05, pages 351–354, 2005. 1, 2
- [11] R. Irony, D. Cohen-Or, and D. Lischinski. Colorization by example. In *Eurographics Symp. on Rendering*, volume 2. Citeseer, 2005. 1, 2, 5, 6, 7
- [12] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012. 1, 4
- [13] A. Levin, D. Lischinski, and Y. Weiss. Colorization using optimization. In *ACM SIGGRAPH 2004 Papers*, pages 689–694, 2004. 1, 2
- [14] X. Liu, L. Wan, Y. Qu, T.-T. Wong, S. Lin, C.-S. Leung, and P.-A. Heng. Intrinsic colorization. In *TOG*, volume 27, page 152. ACM, 2008. 1, 2, 5
- [15] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. *arXiv preprint arXiv:1411.4038*, 2014. 5
- [16] 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*, EGSR'07, pages 309–320, 2007. 1, 2
- [17] W. Ouyang and X. Wang. Joint deep learning for pedestrian detection. In *ICCV*, pages 2056–2063. IEEE, 2013. 1
- [18] G. Patterson and J. Hays. Sun attribute database: Discovering, annotating, and recognizing scene attributes. In *CVPR*, pages 2751–2758. IEEE, 2012. 7
- [19] G. Petschnigg, M. Agrawala, H. Hoppe, R. Szeliski, M. Cohen, and K. Toyama. Digital photography with flash and no-flash image pairs. *ToG*, 2004. 6
- [20] Y. Qu, T.-T. Wong, and P.-A. Heng. Manga colorization. In *ACM SIGGRAPH 2006 Papers*, SIGGRAPH '06, pages 1214–1220, 2006. 1, 2
- [21] E. Reinhard, M. Ashikhmin, B. Gooch, and P. Shirley. Color transfer between images. *IEEE Comput. Graph. Appl.*, 21(5):34–41, 2001. 2
- [22] E. Tola, V. Lepetit, and P. Fua. A fast local descriptor for dense matching. In *CVPR*, pages 1–8. IEEE, 2008. 4, 7
- [23] T. Welsh, M. Ashikhmin, and K. Mueller. Transferring color to greyscale images. *ACM Trans. Graph.*, 21(3):277–280, July 2002. 1, 2, 6, 7
- [24] Z. Yan, H. Zhang, B. Wang, S. Paris, and Y. Yu. Automatic photo adjustment using deep learning. *arXiv preprint arXiv:1412.7725*, 2014. 1, 2, 5
- [25] L. Yatziv and G. Sapiro. Fast image and video colorization using chrominance blending. *Trans. Img. Proc.*, 15(5):1120–1129, 2006. 1, 2
- [26] X. Zeng, W. Ouyang, and X. Wang. Multi-stage contextual deep learning for pedestrian detection. In *ICCV*, pages 121–128. IEEE, 2013. 1
- [27] S. Zheng, A. Yuille, and Z. Tu. Detecting object boundaries using low-, mid-, and high-level information. *CVIU*, 114(10):1055–1067, 2010. 2, 5