Selective Region-based Photo Color Adjustment for Graphic Designs

NANXUAN ZHAO, Harvard University, USA and City University of Hong Kong, Hong Kong
QUANLONG ZHENG, City University of Hong Kong, Hong Kong
JING LIAO, City University of Hong Kong, Hong Kong
YING CAO, City University of Hong Kong, Hong Kong
HANSPETER PFISTER, Harvard University, USA
RYNSON W.H. LAU, City University of Hong Kong, Hong Kong

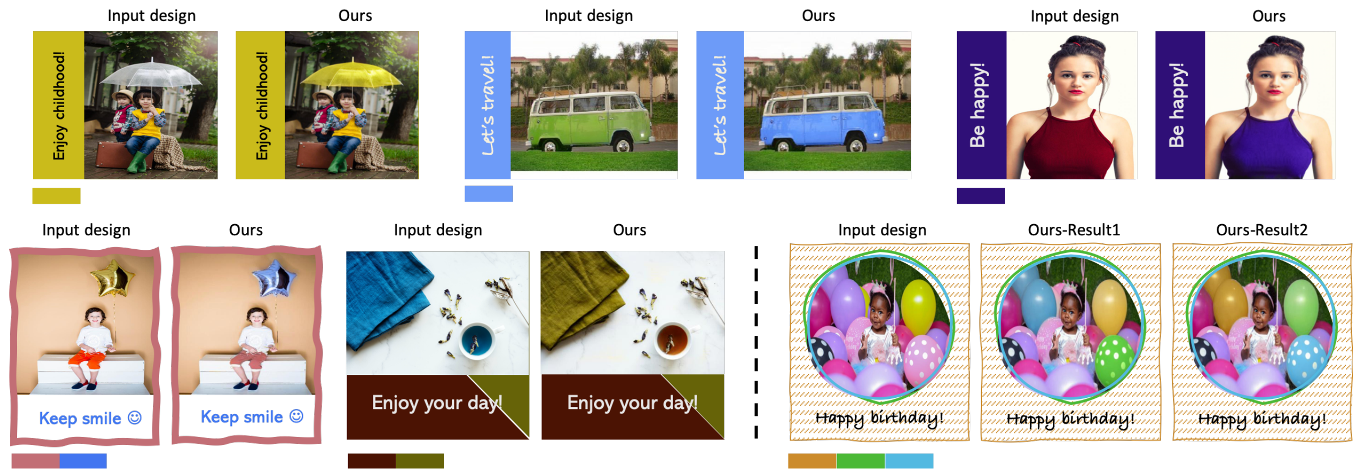


Fig. 1. Photo color adjustment results in the context of graphic designs. When inserting a photo into a graphic design (see the input designs on the left), our model can automatically predict modifiable regions and recolor these regions with the target colors (see the color bars at the bottom of the input designs) to form the output design (see our designs on the right). We show results generated by our model with a single target color in the first row and with multiple target colors in the second row. We can see that our method can suggest appropriate regions for recoloring to the target colors, such that the resulting images still look natural with the original object semantics preserved and the resulting designs look visually more harmonious. Our model is also able to provide multiple suggestions for the user to choose from (see the right-most example in the bottom row).

When adding a photo onto a graphic design, professional graphic designers often adjust its colors based on some target colors obtained from the brand or product to make the entire design more memorable to audiences and establish a consistent brand identity. However, adjusting the colors of a photo in the context of a graphic design is a difficult task, with two major challenges: (1) Locality: the color is often adjusted locally to preserve the semantics and atmosphere of the original image; (2) Naturalness: the modified region needs to be carefully chosen and recolored to obtain a semantically valid and visually natural result. To address these challenges, we propose a learning-based approach to photo color adjustment for graphic designs, which maps an input photo along with the target colors to a recolored result. Our method decomposes the color adjustment process into two successive stages: modifiable region selection and target color propagation. The first stage aims to solve the core, challenging problem of which local image region(s) should be adjusted, which requires not only a common sense of colors appearing in our visual world but also understanding of subtle visual design heuristics. To this end, we capitalize on both natural photos and graphic designs to train a region selection network, which detects the most likely regions to be adjusted to the target colors. The second stage trains a recoloring network to naturally propagate the target colors in the detected regions. Through extensive experiments and a user study, we demonstrate the effectiveness of our selective region-based photo recoloring framework.

CCS Concepts: • **Computer systems organization** → **Embedded systems**; *Redundancy*; Robotics; • **Networks** → Network reliability.

Additional Key Words and Phrases: local color adjustment, recoloring, graphic design

ACM Reference Format:
Nanxuan Zhao, Quanlong Zheng, Jing Liao, Ying Cao, Hanspeter Pfister, and Rynson W.H. Lau. 2021. Selective Region-based Photo Color Adjustment

Authors' addresses: Nanxuan Zhao, Harvard University, USA, City University of Hong Kong, Hong Kong, nanxuanzhao@gmail.com; Quanlong Zheng, City University of Hong Kong, Hong Kong, xiaolong921001@gmail.com; Jing Liao, City University of Hong Kong, Hong Kong, jingliao@cityu.edu.hk; Ying Cao, City University of Hong Kong, Hong Kong, caoying59@gmail.com; Hanspeter Pfister, Harvard University, USA, pfister@seas.harvard.edu; Rynson W.H. Lau, City University of Hong Kong, Hong Kong, rynson.lau@cityu.edu.hk.

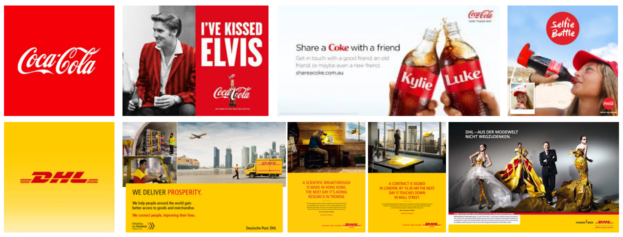


Fig. 2. Graphic design examples where the brand colors are used to color some selected regions in the images.

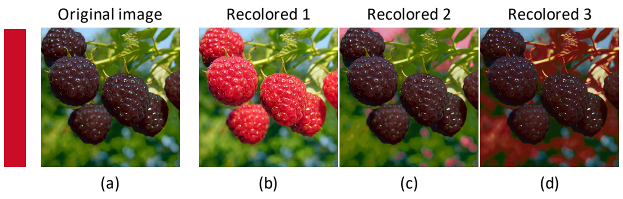


Fig. 3. The importance of selecting suitable regions for recoloring. In this example, changing the mulberry regions to red produces a visually appealing and natural recoloring result (b). Recoloring an inappropriate region may generate implausible and unpleasant results, as shown in (c) and (d).

for Graphic Designs. *ACM Trans. Graph.* 37, 4, Article 111 (August 2021), 17 pages.

1 INTRODUCTION

Photos play an important role in many types of graphic designs, such as posters, advertisements, postcards, and slides. They are not only decorations but also hooks to draw viewers in. When adding a photo into a design, professional designers often need to deliberately edit the colors of the photo according to some target colors. The target color can be the background theme color of the design, or a distinctive brand color to be marketed through the design. For example, the cloth of the man (1^{st} row of Figure 2) is intentionally colored in red to match the red in the Coca Cola logo. The airplane, wall, envelope, and clothes (2^{nd} row of Figure 2) are colored in orange to match the orange in the DHL logo. Such color adjustment helps make the design more memorable to the audience, or establish a consistent brand identity [Ruzzier and De Chernatony 2013; Underwood 2003; Velarde 2018].

However, adjusting the color of a photo in the context of a graphic design is a non-trivial task that poses two unique challenges: 1) **Locality**: instead of updating all the pixels of a selected color in an image by a global operation, graphic designers often prefer to restrict the editing to some local regions in order to preserve the semantics and atmosphere of the image as much as possible. 2) **Naturalness**: rather than altering the color of an arbitrary region, graphic designers need to make careful decisions regarding which region to modify such that the results are semantically valid and visually natural. Improperly selecting a region may end up with an implausible recolored result (Figure 3). Besides, instead of trivially copying the target color to a selected local region, the color of the region needs to be tweaked carefully to render the recolored image to be holistically natural.

There is already a large body of existing works on recoloring a photo according to given colors. They can fall into two broad categories: palette-based approaches and stroke-based approaches. Unfortunately, none of them can be used to address the aforementioned challenges of the color adjustment problem in the context of a graphic design. The palette-based approaches [Chang et al. 2015; Tan et al. 2018] recolor an input image by editing the color palette extracted from the image. However, it is not clear what is an effective strategy to map the target colors to the colors in the palette. More fundamentally, these methods determine which pixels to change purely based on low-level relationship between the colors in the image and those in the palette, making it difficult to restrict the changes to local semantically meaningful regions. The stroke-based approaches [Endo et al. 2016; Levin et al. 2004; Luan et al. 2007; Yatziv and Sapiro 2006] can limit color changes to a local region, but rely on users to indicate which local region to modify by putting scribbles on it, which may sometimes require intensive strokes to obtain a good result.

In this paper, we propose a novel deep learning based approach to photo color adjustment in graphic designs to tackle the two challenges mentioned above in a unified framework. Given an input image and one or more target colors specified by users, our approach automatically selects local regions in the input image for each of the target color to perform color adjustment, with an objective of producing a visually natural recolored image. Our approach consists of two stages.

In the first stage, we train a network to select modifiable regions for recoloring by jointly leveraging natural prior and design knowledge. Our insight is that both natural images and graphic designs are promising sources to extract useful knowledge for region selection in our recoloring task. Hence, we first learn a natural prior from a large natural image dataset to guide region selection based on the color statistics in our natural world. However, using the learned natural prior alone is not sufficient to constrain the region selection problem, since a single image may have multiple regions whose color changes can induce equally plausible recoloring results. This calls for adding more rules to regularize the region selection process. However, hand-engineering some heuristics requires a lots of manual efforts and expert knowledge. Instead, we propose to directly learn the knowledge of how professional designers choose regions for recoloring from a collection of graphic designs. To this end, we have collected a new labeled graphic design dataset, where target colors along with their corresponding regions on the designs can be intuitively identified.

In the second stage, we learn a recoloring network to adjust the color of the selected region, while still preserving the visual naturalness of the entire image. In order to produce a more vivid and natural recolored image without artifacts near the region boundary, we introduce a new deep learning recoloring model based on generative adversarial network (GAN) [Goodfellow et al. 2014] and a soft boundary propagation method. We have conducted extensive qualitative and quantitative evaluations. Results show that our method outperforms existing recoloring methods for the recoloring task in the context of graphic design.

In summary, we make the following contributions:

- We propose a deep learning based approach to address the photo color adjustment problem in graphic designs, which can selectively adjust local regions in a color photo according to given target colors.
- We propose a model to localize modifiable image regions based on the target colors, by taking advantage of both natural prior and design knowledge that are learned end-to-end from data without using any heuristics.
- We have collected a new graphic design dataset, containing about 2000 images used in real-word graphic designs, each of which is annotated with human-identified target colors and corresponding modified regions.

2 RELATED WORK

Our work is related to an earlier work that also tries to automatically select regions for recoloring [Cohen-Or et al. 2006]. While their goal is to achieve color harmonization of the whole image, ours is to recolor based on the given target colors. In this section, we provide an overview of the major related works on example-based and palette-based recoloring that globally change the colors, and stroke-based methods that locally adjust the colors via user interactions.

2.1 Example-based Recoloring

Example-based recoloring aims to transfer the color style of a reference image onto an input image. Chang *et al.* [2005] categorize the color space through psychological experiments, and transfer the color of matching pixels within the same category. HaCohen *et al.* [2013] propose a method to automatically propagate image enhancement operations to the input image with a similar content to the reference image, based on dense correspondences. Huang *et al.* [Huang et al. 2014] build a factor graph to learn the color distribution from examples, and generate new recolored images using Markov chain Monte Carlo (MCMC) methods to sample in the space. Recently, deep learning techniques are widely used in example-based recoloring [Gatys et al. 2016; He et al. 2018; Luan et al. 2017]. Luan *et al.* [2017] introduce local affine constraints on [Gatys et al. 2016] to achieve photo-realistic color transfer. Example-based recoloring often requires a reliable correspondence between the reference image and the target image. To mitigate this problem, Liao *et al.* [2017] present a deep learning approach to build dense semantic correspondences between two images in a hierarchical way and then transfer colors accordingly. While these works can generate impressive results, they are not aligned with our problem setting, as they transfer colors to the input image without maintaining any original color statistics and require an additional reference image as input.

2.2 Palette-based Recoloring

In palette-based recoloring, it is critical to address the palette extraction problem and the image decomposition problem based on the extracted palette. Chang *et al.* [2015] propose an intuitive interaction tool for fast exploration of recoloring results by manipulating a palette extracted using a variant of k-means. Zhang *et al.* [Zhang et al. 2017a] decompose colors of the entire image into linear combinations of basis colors in the palette. Aksoy *et al.* [2017] propose a soft color segmentation method to decompose an image into color layers with alpha channels. Tan *et al.* [2018] decompose images in the RGBXY-space, which is more efficient. All these methods, however, require users to manually assign color correspondences between the input and target palettes. Wang et al. [2010] learns a prior of texture-color relationship over possible recoloring from a large data-set. This allows users to enhance the color of an image based on example color themes and maintain natural look of recolored images. A recent work [Kim and Suk 2018] aims to automatically adjust the colors of the image in a graphic design based on a single target color. It is heuristic-based by changing the color of a region that has the closest color to the target one. This method can easily generate unnatural results without considering any semantic constraints. Instead, our model can automatically localize semantically meaningful regions to be recolored so that the resulting images still look plausible and natural.

2.3 User-guided Recoloring

This class of methods propose to recolor images by propagating a desired color through user scribbles [Endo et al. 2016; Levin et al. 2004; Li et al. 2008; Luan et al. 2007; Yatziv and Sapiro 2006]. Levin *et al.* [2004] propagate scribbles to pixels with similar luminance. Yatziv and Sapiro [2006] further accelerate the process by using luminance-weighted chrominance blending and fast intrinsic distance. Luan *et al.* [2007] present an interactive recoloring framework for natural images with complex texture by incorporating both intensity-continuity and texture-similarity constraints to group pixels into coherent regions. Endo *et al.* [2016] train a CNN to learn visual features automatically for color propagation. All these methods require users to specify, often intensive, strokes or points to indicate the regions to modify. Instead, our method automatically predicts the regions that are suitable for recoloring, rendering the recoloring task to be easier and more efficient.

3 OVERVIEW

Given a graphic design and a photo to be inserted into the design, we want to construct a model to adjust the color of the photo locally, so that the recolored photo matches with the given target colors and still looks natural. We focus our study on common graphic designs that contain a small number of visual elements, such as posters, advertisements, and fliers, where the target colors are often clearly defined and can be extracted easily. We extract the target colors from an input design by simply using the major colors of other existing elements. We choose to train a convolutional neural network (CNN) to adjust colors based on low-level visual cues and high-level semantics in the input image.

Our system carries out the color adjustment task in two stages (Figure 4). The first stage is to find out which parts of an image are suitable to be changed to a given target color. We train a **Region Selection Network** to find possible regions, and rank them based on their likelihoods so that users may optionally choose to customize their results. In the second stage, our **Recoloring Network** learns to propagate each target color naturally in its corresponding selected

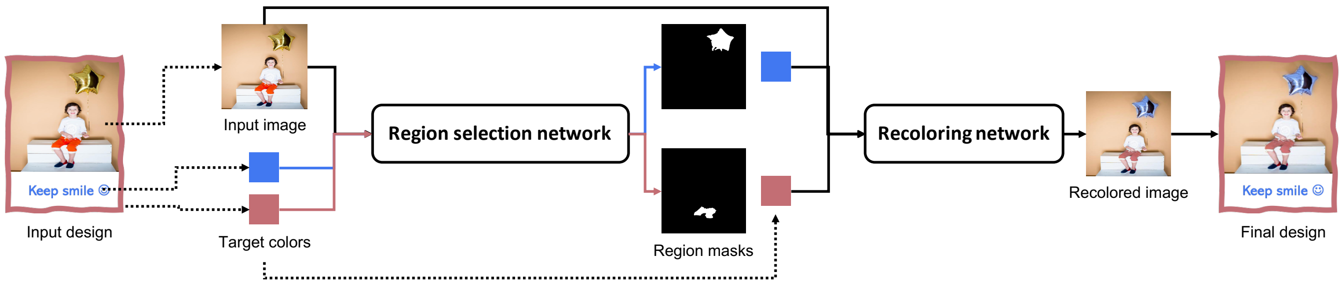


Fig. 4. Overview of our framework. Our framework contains two stages: the **region selection network** aims to find modifiable regions of the original image for color adjustment; the **recoloring network** aims to propagate the target colors in their corresponding selected regions naturally.

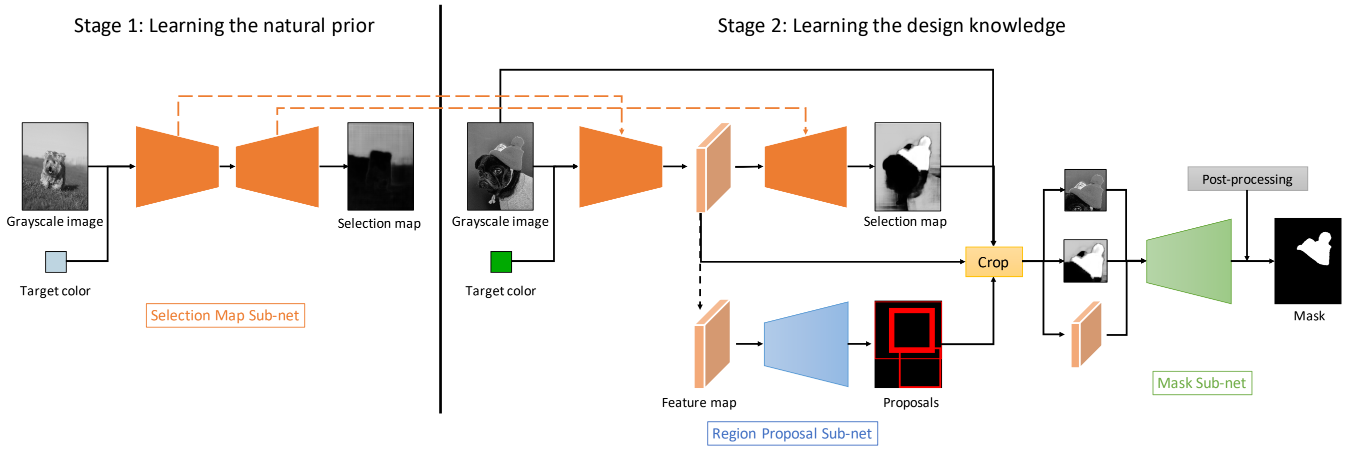


Fig. 5. Our region selection network. Given an input image and a target color, our model is able to automatically predict a set of modifiable regions in a binary mask format. Our region selection model has two stages. The first stage learns the natural prior from a large natural image dataset to predict a pixel-wise selection map indicating the likelihood of each pixel to be recolored to the target color. The second stage learns the design knowledge from a newly collected graphic design dataset to further constrain the region selection task.

regions. One merit of such a two-stage framework is that it enables users to easily express their preference in the recoloring process. To do this, instead of automatically identifying a single modifiable region for a target color, our region selection network can produce multiple region suggestions ranked by their likelihoods. Users are able to select their preferred region among all the suggestions to customize their own recoloring results.

Collecting a large-scale paired data (i.e., images before and after recoloring) for training is difficult. Fortunately, we can collect a large amount of colorful, natural and aesthetical images from online repositories. Our idea here is to consider these collected color images as the recolored results, such that target colors are directly accessible (i.e., on the color images) and the modified regions with respect to the target colors can be identified either automatically or via intuitive manual labeling. However, we still need to obtain the original images (i.e., the images before recoloring), and the original colors of the modified regions are unknown and hallucinating the original colors is not an easy task. Even using the state-of-the-art colorization model [Zhang et al. 2017b], the generated color distribution still has a big gap to the real color distribution of natural images, which can mis-train the model. To address this problem, in the Region Selection Network, we propose to use grayscale images as input, which forces the model to focus more on the context information than on the original colors of the modifiable regions. We demonstrate in Section 6.3 that our model is able to locate suitable regions for a given target color, from a grayscale image. Further in the Recoloring Network, we cast color propagation as an inpainting problem, which will be discussed in Section 5.

4 REGION SELECTION NETWORK

Given an input image I and a target color C, the region selection network $\mathcal{F}$ aims to predict a set of modifiable regions for the target color. Figure 5 shows the pipeline of our region selection model $\mathcal{F}$. It consists of three sub-nets, learned from two kinds of data sources: a natural image dataset (**Natural Prior**) and a graphic design dataset (**Design Knowledge**). The selection map sub-net $\mathcal{S}$ (Figure 5, left) predicts a selection map, which indicates the likelihood of each pixel to be recolored to a given target color. We train S on a large natural image dataset (i.e., ImageNet [Russakovsky et al. 2015]) to learn a prior on color patterns of natural objects and apply it during inference to estimate modifiable regions. This is because when choosing a region for recoloring, we tend to exploit our natural color priors of different real-world objects to make the decision. Particularly, if changing the color of a region to a target color will change its semantic class or even make it semantically implausible (e.g., changing the sky to green), the region is less likely to be selected.

While our learned natural prior can help enforce the naturalness of the recolored images by suggesting probable regions, it does not take into account design heuristics that expert designers usually use in selecting regions in real-world recoloring tasks. Modeling the knowledge of professional designers can help resolve certain ambiguities in region selection, especially when there are multiple regions with similar likelihoods of being selected. For example, both the wall and t-shirt in an image can be recolored as red according to the natural prior. However, graphic designers may choose to recolor the t-shirt as it is a more salient region and thus can catch the readers' attention better. To this end, we have collected a labeled graphic design dataset and trained our network on it. However, region selection by human experts can be inherently subjective. Hence, when learning design knowledge, instead of predicting a single selection map, we augment the selection map sub-net with a region proposal sub-net and a mask sub-net to predict a set of candidate selection maps. Note that in our model, design knowledge learning and natural prior learning share the same selection map sub-net, which means that the selection map sub-net is first pre-trained on natural images and then fine-tuned on graphic designs. This allows our model to capture both knowledge jointly.

Specifically, our region selection network $\mathcal{F}$ takes as input a grayscale image I_L (the L channel of the input image in the CIE Lab color space), and the chrominance values (ab color channels) of the target color C_{ab}, and outputs a set of modifiable regions that are suitable to be recolored to C_{ab}. Each region is represented by a mask m_i along with a confidence score v_i indicating its the likelihood of being selected:

$$(m_i, v_i)_{i=1,\ldots,n} = \mathcal{F}(I_L, C_{ab}), \tag{1}$$

where n is the number of proposed regions. Although our model $\mathcal{F}$ only takes one target color C as input, it can be easily extended to take multiple target colors, by running it multiple times, each with a different color as input.

4.1 Learning the Natural Prior

Given a grayscale image $I_L \in R^{H \times W}$ and the chrominance values of a target color $C_{ab} \in R^{H \times W \times 2}$, our selection map sub-net $\mathcal{S}$ predicts a per-pixel selection map $S \in R^{H \times W}$ for I_L as:

$$S = \mathcal{S}(I_L, C_{ab}), \tag{2}$$

where a higher value indicates a higher probability for the pixel to be selected and recolored to C_{ab}. The selection map needs to be content-aware and conform to the natural color statistics. Therefore, we propose to learn the selection map sub-net from a collection of natural images that inherently come with statitsics of natural colors.

To train the network, we use ImageNet [Russakovsky et al. 2015], one of the largest public natural image datasets, and segment each image in it into a set of disjoint segments. Since the target colors and ground truth selection maps are unknown a priori on these images, we synthetically generate our training data. In particular, for each image I in the dataset, we randomly select an image segment and use its mean color as the target color C_k for the image. Given the target color, we assume that the pixels in the selected segments have a higher chance to be colored to C_k than all the other pixels in I, and thus create a binary mask for the selected segment as the selection map $\hat{S_k}$. We end up with the training data in the form of $(I, C_k, \hat{S_k})$.

While there are already many available semantic segmentation datasets, containing well-annotated image segmentation, they are not applicable to our task, as each semantic segment may consist of multiple regions with different color distributions. For example, a person is often labeled as one semantic segment, but the t-shirt of the person may have a more dynamic color range than the skin. Besides, existing semantic segmentation datasets cover only a limited number of object types. Instead, traditional image segmentation methods that group pixels of similar colors into regions are more suitable for our problem. Thus, we obtain our image segmentations using a state-of-the-art image segmentation method, Single-scale Combinatorial Grouping (SCG), [Arbeláez et al. 2014] on our dataset. SCG is a bottom-up method with an efficient normalized cut algorithm and grouping method, designed to generate image segmentation and object candidates. The scale of SCG is set to 0.6 to obtain a mid-level segmentation.

For each training tuple $(I, C_k, \hat{S_k})$, we train the selection map sub-net with the following pixel-wise cross entropy loss:

$$L_{np} = -\sum_x \sum_y [S_k^{\hat{x,y}} \log S_k^{x,y} + (1 - S_k^{\hat{x,y}}) \log(1 - S_k^{x,y})]. \tag{3}$$

The initial selection map derived from the natural prior is a coarse probability map, and will be refined by incorporating the design knowledge, which is described in the next section.

4.2 Learning the Design Knowledge

To incorporate the design knowledge into region selection, we propose to learn from the graphic design dataset. This is because when adding a photo into a graphic design, professional designers often deliberately edit the color of the photo to match with either the brand color or the background theme color. For example, the cloth of the man in Figure 2 (1^{st} row) is intentionally colored as red to echo with the main color of the Coca Cola logo. The airplane, wall, envelope, and clothes in Figure 2 (2^{nd} row) are colored as orange-yellow to echo with the main color of the DHL logo. Such color adjustment helps make the design more memorable to the audience, or establish a consistent brand identity. To capture this knowledge and deal with the multi-modal nature of predicting human choices, we build a region selection model by augmenting the selection map sub-net with a region proposal sub-net and a mask sub-net, as shown in Figure 7. Given a grayscale image I_L, with the chrominance values of a target color C_{ab}, the region selection model outputs a set of masks $\{m_1, m_2, ..., m_n\}$ with confidence scores $\{v_1, v_2, ..., v_n\}$, indicating a list of suggested regions that are suitable to be changed to C_{ab}.

4.2.1 Training Data. We downloaded around 20,000 designs with various appearances from *Canva*[1], a professional online design repository. After removing duplicate or low-resolution designs, we

[1] *https://www.canva.com/templates/*

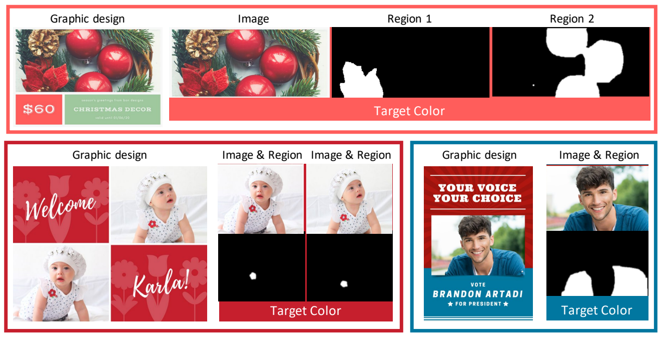


Fig. 6. Examples from our graphic design dataset. For each design, we extract its target color and photos. For each photo, the mask of the regions that use the target color is labeled.

further filtered them by manually selecting the designs where the colors of some regions are consistent with its theme colors. We hired two people with graphic design experience to annotate each design by pointing out its theme colors and specifying the masks of the corresponding regions in the photo. Note that there may be more than one region to be annotated for a single color since designers likely modified multiple regions. If multiple regions of different semantics are selected for a single color, we label the regions in seperate masks (as shown at the first row of Figure 6). If multiple disjoint regions belonging to the same semantic part are selected, we label them in a single mask (e.g., as shown at the lower right part of Figure 6). The theme colors (i.e., target colors) are often from the background colors or the text/brand colors on the designs. Finally, we obtained 1,968 pairs with 2,876 individual region masks. We randomly selected 1876 paris for training and use the rest for testing. Figure 6 shows several examples in our graphic design dataset.

4.2.2 Loss Function. The selection map sub-net still predicts a selection map S from I_L and C_{ab}. The region proposal sub-net $\mathcal{P}$ takes as input an intermediate representation U from the selection map sub-net and generates a set of region proposals (i.e., bounding boxes):

$$\{B_i, v_i\} = \mathcal{P}(I_L, C_{ab}),\ B_i = [x_i, y_i, w_i, h_i],\ v_i \in [0,1],$$

where the i-th bounding box B_i is represented by its center (x_i, y_i), width w_i and height h_i, and confidence score v_i. For each region proposal B_i, we use its bounding box to crop the corresponding regions on I_L, S and U, obtaining the cropped results $I_L^{B_i}$, S^{B_i} and U^{B_i}, which are then concatenated and sent into the mask sub-net. The mask sub-net $\mathcal{M}$ generates a mask for the region:

$$m_i = \mathcal{M}(I_L^{B_i}, S^{B_i}, U^{B_i}).$$

Region Proposal Sub-net. Here, we borrow the idea from the state-of-the-art object detectors [Redmon and Farhadi 2018; Ren et al. 2015], and predict bounding boxes by regressing offsets from a set of anchor boxes. In particular, we project each point on the input feature map U back onto the image plane and consider a set of K anchor boxes of different aspect ratios. For each anchor box, we predict a confidence score and four scalars that transform the anchor box to an output bounding box. In our implemention, we set $K = 4$ and estimate the anchor boxes as cluster centroids after clustering all the boxes in the graphic design dataset using K-means.

Given an anchor box with center position (x', y'), width w' and height h', we predict an offset vector (t_x, t_y, t_w, t_h). The predicted bounding box is then defined as:

$$\begin{aligned} x_i &= sigmoid(t_x) + x', \\ y_i &= sigmoid(t_y) + y', \\ w_i &= w' e^{t_w}, \\ h_i &= h' e^{t_h}. \end{aligned} \tag{4}$$

During training, we follow [Redmon and Farhadi 2018] to classify each anchor box into three different categories: (1) leading anchor box that has the largest overlapping area with the ground truth region; (2) positive anchor box that has IoU larger than 0.5 with the ground truth region and is not a leading anchor box; (3) negative anchor box that does not belong to (1) or (2).

We denote leading, positive and negative anchor boxes as $B_\#$, B_+, B_-, respectively. The loss function is defined as:

$$L_{bbx} = \sum_{i \in B_\# \cup B_-} L_{bce}(v_i, \hat{v}_i) + \sum_{i \in B_\# \cup B_+} ||T_i - \hat{T}_i||_2^2, \tag{5}$$

where L_{bce} is a binary cross entropy function. v_i and T_i are predicted confidence score and offset vector for the i-th anchor box, respectively, and $\hat{v}_i$ and $\hat{T}_i$ are the ground truth. For $B_\#$, we set their ground truth confidence scores as 1. For B_-, we set their ground truth confidence scores as 0.

Mask Sub-net. For each region proposal B_i, the mask sub-net outputs a residual, which is added to the cropped region from the selection map S_{B_i} to get the final predicted mask M. We choose to predict a residual since S_{B_i} is already a coarse shape of the selected region. The mask sub-net is trained to approach the ground mask $\hat{M}$ in a binary cross entropy sense:

$$L_{mask} = -\sum_x \sum_y [\hat{M}_{x,y} \log M_{x,y} + (1 - \hat{M}_{x,y}) \log(1 - M_{x,y})]. \tag{6}$$

4.3 Network Architectures

Figure 7 shows the network architecture of our region selection model.

Selection Map Sub-net. We use a U-Net encoder-decoder architecture with skip connections [Ronneberger et al. 2015], as it has shown good performance in many image manipulation tasks [Portenier et al. 2018; Zhang et al. 2018, 2017b]. We represent an input target color as a two-channel color map with the same resolution as the input image, by duplicating its ab values spatially. The selection map sub-net consists of 10 convolutional blocks, each of which contains 2~3 *conv-relu* pairs, followed by a batch normalization layer [Ioffe and Szegedy 2015]. Skip connections are added between the outputs of 1st and 9th, 2nd and 8th, and 3rd and 7th blocks.

Region Proposal Sub-net. We choose to predict region proposals on feature maps, rather than in pixel space, in order to capture object-level information better. Thus, the input to the region proposal sub-net is the output of the "conv7_4" layer of the selection

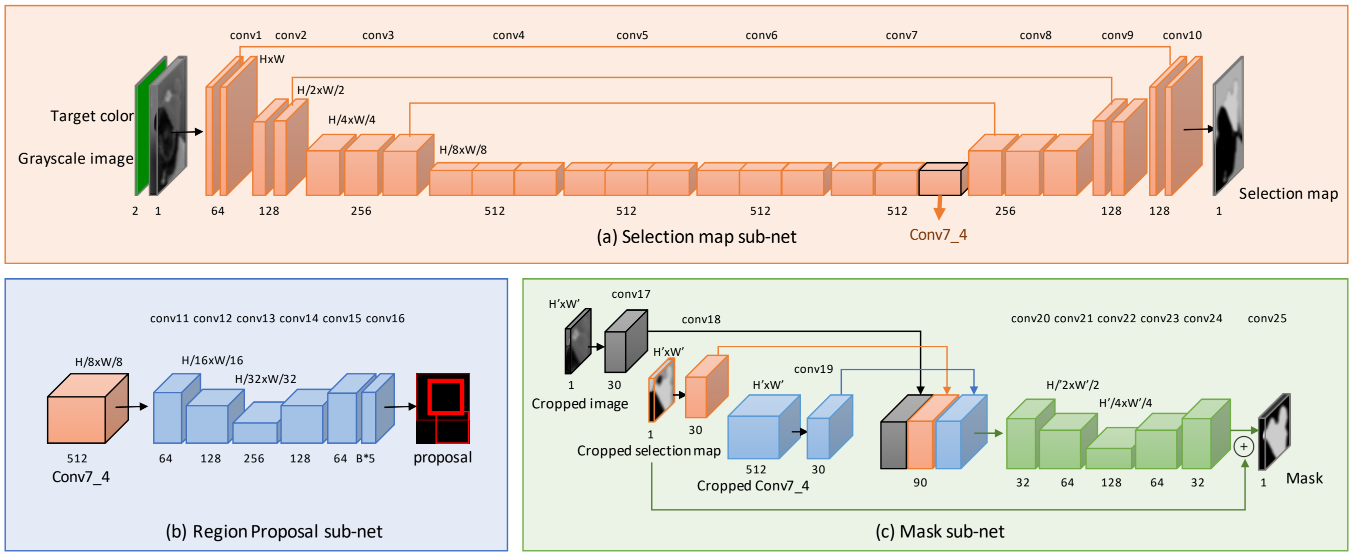


Fig. 7. The network architectures of modules in our region selection model.

map sub-net, just before the upsampling operation. The region proposal sub-net includes 6 convolutional layers, each followed by a batch normalization layer and a ReLU activation function, except for the last convolutional layer. The output of the last convolutional layer contains the offsets and confidence scores for all anchor boxes. The confidence scores are normalized by a sigmoid function into [0,1].

Mask Sub-net. It takes three inputs: the cropped regions of the selection map, input image, and feature map (i.e., "conv7_4") based on a region proposal. They are first rescaled to the same resolution using bilinear interpolation and then passed through three separate convolutional layers, respectively, to obtain three feature maps. The feature maps are then concatenated along channel, and fed into a network composed of five convolutional layers with batch normalization and ReLU nonlinearity to produce a residual map. The residual map is added to the input cropped selection map to generate an output mask.

4.4 Training

We train the whole region selection model in three stages.

S1: We train the selection map sub-net $\mathcal{S}$ on synthetic data generated from the natural image dataset (i.e., ImageNet) using Eq. 3.

S2: We take $\mathcal{S}$ trained in **S1** as a pre-trained model. Then, while keeping the weights of the region proposal sub-net P, we jointly train the selection map sub-net $\mathcal{S}$ and mask sub-net $\mathcal{M}$ on the graphic design dataset using a combined loss function $\lambda_{np}L_{np} + \lambda_{mask}L_{mask}$. This fine-tuning step helps the model and the feature map (i.e., "conv7_4") capture both natural prior and design knowledge, which will boost the performance of predicting region proposals (as validated in Section 6.3.4). We use the ground truth bounding boxes to create inputs to the mask sub-net.

S3: After the first two stages, we fix the weights of $\mathcal{S}$ and only train the region proposal sub-net $\mathcal{P}$ on the graphic design dataset.

We use the Adam optimizer [Kingma and Ba 2014] with $lr = 1e{-}4$ for training. Training images are first resized to 256×256 and then randomly cropped to 224×224. As all our models are fully convolutional, they support arbitrary input size during inference. In addition, our region selection model can be used in an end-to-end manner during inference. For **S1**, we train for 15 epochs with a batch size of 100, and the training procedure takes around 5 days on 6 × Titan XP GPUs. For **S2** and **S3**, we train a batch size of 10 on 2 Titan XP GPUs for 50 epochs. For other parameters, we set the size of the cropped image as $W' = H' = 56$, and the weight parameters of different loss functions as $\lambda_{np} = \lambda_{mask} = \lambda_{bbx} = 1$ for L_{np}, L_{mask}, and L_{bbx}, respectively.

5 RECOLORING NETWORK

Given a selected region in an input color image and a target color, the recoloring network aims to propagate the target color into selected region naturally. Since ground truth recolored images are not readily available and there are weak correlations (if any) between the target colors and the original colors of their corresponding regions, we choose to mask out the colors within the selected region of the input image. In this way, our task can be viewed as a color inpainting problem (i.e., fill in the color of a missing region according to a target color). This formulation also allows us to learn the network from synthetic training data as detailed below. Specifically, the inputs to the recoloring network include an input image $I \in R^{H \times W \times 3}$, masked by a selected region mask $M \in \{0,1\}^{H \times W}$, and a target color map $C \in R^{H \times W \times 3}$, where the selected region is filled with the corresponding target color. In M, we set the values inside the selected region as 1 and 0 elsewhere. The output of this network is a residual chrominance map $\Delta I_{ab} \in R^{H \times W \times 2}$ to be added to the ab channels of image I to get the chrominance values of the final recolored image R_{ab}.

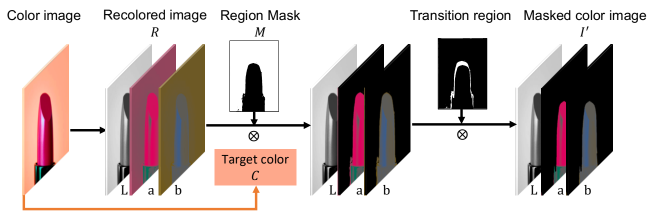


Fig. 8. Dataset construction for our recoloring network. We generate the synthetic dataset by taking a color image as the recolored result, and masking out the color of a region (i.e., the background in this example) to generate an input masked color image. The target color is the mean color of the masked-out region in the color image.

5.1 Dataset Construction

Training our network requires a large scale dataset of image pairs before and after recoloring, which is not available and is difficult to obtain. Asking designers to manually edit images is tedious, time-consuming, and may also limit the diversity of the generated pairs. We bypass this problem by generating a synthetic dataset to simulate diverse inputs that the network may encounter during inference. In particular, as illustrated in Figure 8, given a color image, we directly take its ab color channels as the output recolored image R_{ab}. To generate the corresponding inputs, we randomly sample one region in the segmentation map (obtained using SCG [Arbeláez et al. 2014]). We use the region mask M to indicate the missing region, and we use its mean color as the target color C. We then mask out the colors of the pixels in the selected region by setting their chrominance values to 0 to obtain a masked color image I'_{ab}. The network is then tasked with reconstructing R_{ab} by taking C, M and I'_{ab} as input, and thus learns how to propagate the target color C within the region M so that the predicted color will be coherent with the context of the region in I'_{ab}.

However, many objects involve fuzzy boundaries, such as the hair of a furry dog [Aksoy et al. 2018]. An imperfect binary region mask predicted by our region selection model may produce serious artifacts near the boundary. To address this problem, during the data synthesis process, for each selected region, we create a transition region within a distance $r \sim U[0, 20]$ from the region boundary. The pixels in I'_{ab} that are inside this region are turn into gray (by zeroing out their ab color values). In this way, the network is forced to treat the areas around the region boundaries differently and learn how to propogate colors correctly on both sides of the boundaries. We show the effectiveness of this method in Section 6.4.

5.2 Training

The recoloring network (Figure 9) uses almost the same architecture as the selection map sub-net, except that we further add a skip connection from the input image and the output residual map to form the final output. All the inputs are concatenated along channel and then send into the network. We train the recoloring network on the ImageNet dataset [Russakovsky et al. 2015]. During training, each training example contains only a single selected region. However, during test time, our model can support recoloring multiple regions in a single feed-forward pass, by simply modifying the inputs to encode the information of all the regions.

To train the network, inspired by recent works on image-to-image translation [Iizuka et al. 2017; Isola et al. 2017], two loss functions are jointly used. An l_2 reconstruction loss is used to encourage the network to correctly propagate the target colors within their respective designated regions. A GAN loss is also introduced to further improve the realism of the recoloring results.

We define the l_2 loss as:

$$\mathcal{L}_{MSE} = ||\mathcal{R}(I'_{ab}, C, M) - R_{ab}||^2. \tag{7}$$

For the GAN loss, we use the LSGAN loss [Mao et al. 2017] since it is shown to be more stable during training:

$$\mathcal{L}_{GAN} = \mathbb{E}_R[(D(R) - 1)^2] + \mathbb{E}_{I'_{ab}}[D(\mathcal{R}(I'_{ab}, C, M)^2]. \tag{8}$$

As the training dataset only covers a limited set of target colors, during training, for half of the training examples in each sampled min-batch, we assign their target colors with random colors. Since the ground truth recolored images are not known for such examples, we only apply the GAN loss on them to encourage their recolored images to be as real as possible.

For the discriminator, we use the PatchGAN discriminator [Isola et al. 2017], which is commonly used in recent image manipulation tasks. Rather than classifying the whole image, it tries to classify if each randomly sampled image patch from the generated image is real or fake. This enables our model to generate more realistic local color details and more natural color transition across region boundaries. The discriminator consists of 5 convolutional layers. Each of the layers is followed by a batch normalization layer and LeakyRelu nonlinearity, except the output layer.

The final loss is a linear combination of $\mathcal{L}_{MSE}$ and $\mathcal{L}_{GAN}$ with both weights set to 1. We use the Adam optimizer [Kingma and Ba 2014] with a learning rate of $1e-4$ for training. By default, we train the network for 15 epochs with a batch size of 100. The training procedure takes around 5 days on $6 \times$ Titan XP GPUs.

6 RESULTS AND EXPERIMENTS

We show our automatic recoloring results for various graphic designs in Figure 1 and Figure 10. All the images are not seen by our method during training. For each example, we select the top ranked regions predicted by the region selection network for the given target color. We can see that after adjusting the color of the photos, the photos become more visually harmonious with the graphic designs, while remaining natural. Note that our model is able to provide multiple suggestions for the same inputs in real time. The selected regions span objects of different semantic classes at different scales, and at both foreground and background. We first compare our framework with several existing methods qualitatively and quantitatively, and then conduct experiments to further analyze the different components of our framework.

6.1 Test Set

To evaluate the performance of our framework, we first prepare a test set. We collected a set of 88 photos from an online stock repository (*pexels.com*) and previous recoloring works [Chang et al.

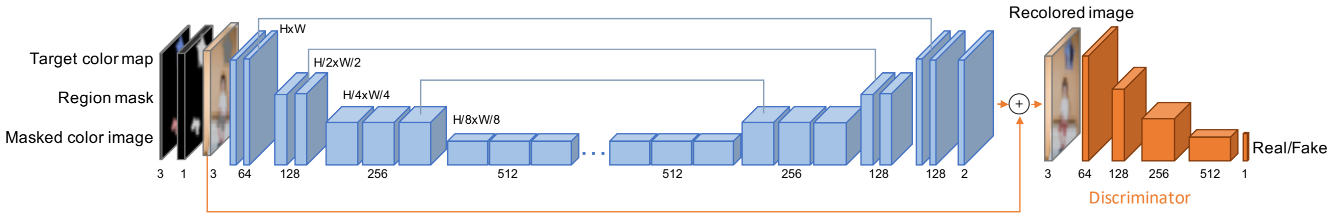


Fig. 9. The architecture of our recoloring network.

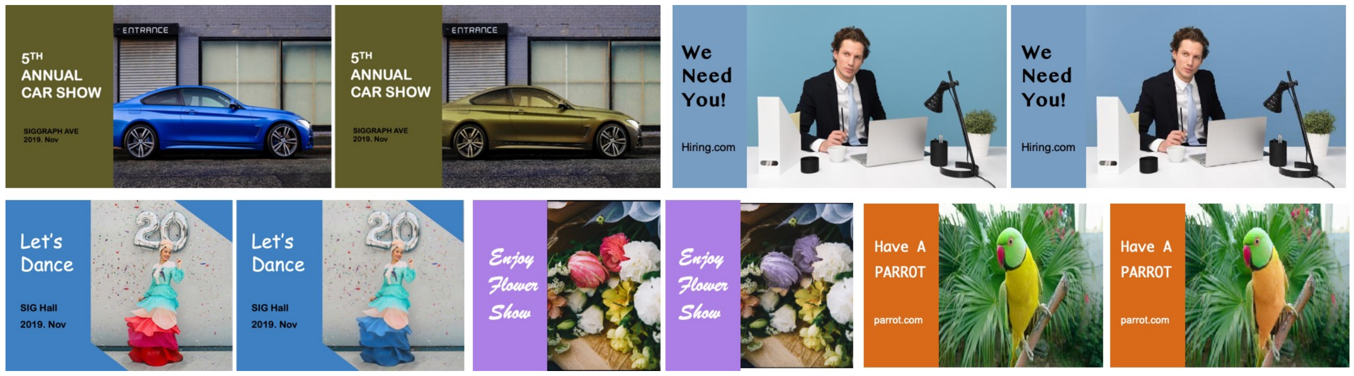


Fig. 10. Automatic recoloring results. We show the input designs on the left and recolored designs on the right. After recoloring, the designs become more visually harmonious with their corresponding target colors. We visualize the target color in a vertical bar on the left of each design.

2015; Kim and Suk 2018; Tan et al. 2018]. The content of the photos ranges from natural scene to indoor scene, from human, animal, food to plant. For each photo, we randomly select one or two target colors forming a set of 136 evaluation cases (i.e., each case contains an input image and a single target color). We randomly split them into three groups, each assigned to a different professional graphic designer. For each design case, the designers were asked to mask a region that they believed to be most suitable to be modified according to the target color and recolor the input image accordingly to maintain a natural looking. They were allowed to use any tools that they wanted, such as photoshop. We take their region selection and recoloring results as the ground truth.

6.2 Comparison to Prior Methods

We compare our method with the following methods:

- A state-of-the art work on photo color adjustment for graphic designs [Kim and Suk 2018]. It adjusts the hue of an input photo iteratively until the minimum hue difference ($\triangle Hue$) between the target color and the colors in the image palette satisfies the condition (i.e., $\triangle Hue > 90^\circ$ or $\triangle Hue < 5^\circ$). This method will not adjust color if the hue difference is too large in order to get good results. They summarize the color palette using K-means.
- Palette-based recoloring methods [Chang et al. 2015; Tan et al. 2018] with nearest neighbor (**NN**) search. Note that these methods are neither intended nor designed for photo recoloring in graphic designs. We use them as baselines for comparison. Palette-based methods require users to manually manipulate the color palette of an input image for recoloring. To automate this process, we follow the approach of [Kim and Suk 2018]. Given a target color, we adjust its nearest color in the original palette to it to get a target palette.
- Palette-based recoloring methods [Chang et al. 2015; Tan et al. 2018] with our region selection (**Reg.**). Rather than modifying the nearest color, we adjust colors in the original palette that dominate the region selected by our method to the target color. It is worth noting that the dominant color can also appear outside our predicted region.

6.2.1 Qualitative Results. We show qualitative results in Figure 11. As can be seen, even though improving visual harmony of the input photos w.r.t the target colors, other methods fail to deliver natural and aesthetically pleasing results. This is mainly because, other methods modify colors solely using pixel-level information (i.e., colors) without exploiting semantic features of the input photos, thereby changing the pixel colors globally and giving unnatural colorizations for some semantic regions (e.g., turning the sky to green in the 2^{nd} column and face to purple in 4^{th} column). In contrast, our method is able to restrict color changes to carefully selected local regions (i.e., the hat in the 1^{st} column, the grass in the 2^{nd} column, bus in 3^{rd} row, and cloth in 4^{th} column) and, therefore, produce natural recolored results.

6.2.2 Quantitative Results. We first calculate the PSNR of different methods on our test set and show the results in Table 1. Our method achieves the highest PSNR score than the prior methods.

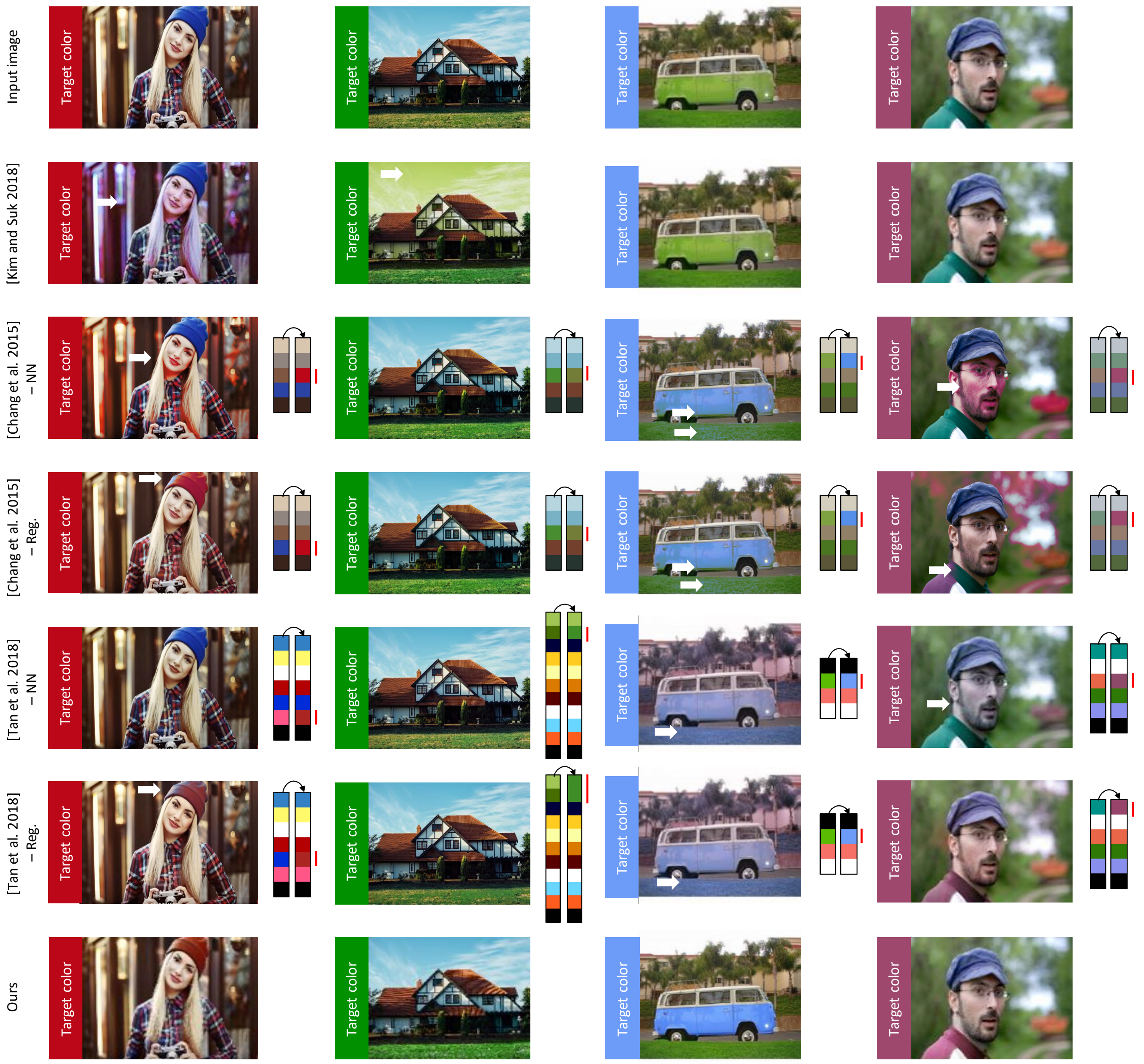


Fig. 11. Qualitative comparison of our method with an existing photo recoloring method for graphic designs [Kim and Suk 2018] and variants of two palette-based methods [Chang et al. 2015; Tan et al. 2018]. For the NN variant, the color in the original palette of an input image that is nearest to a target color is changed to the target color. For the Reg. variant, we use our region selection network to predict a region to change, and modify the colors in the orignal palette that dominate the predicted region to the target color. For result of each palette-based method, we show an original palette (left) and a target palette (right), and highlight the changed colors with red bars. These results show that our method can generate more natural results.

When provided with our region selection results, [Tan et al. 2018]-**Reg.** achieves a higher PSNR score than [Tan et al. 2018]-**NN** and even [Kim and Suk 2018], indicating the effectiveness of our region selection.

User Study. In addition to PSNR, we conduct a user study on 75 design cases randomly selected from the test set on Amazon Mechanical Turk (AMT). Participants were asked to complete a series of pairwise comparisons. For each pair of designs, we asked them to answer two questions: "Which design looks more harmonious?"

Table 1. PSNR scores of different methods on our test set.

Methods	PSNR
[Chang et al. 2015]-NN	20.52
[Chang et al. 2015]-Reg.	21.68
[Tan et al. 2018]-NN	22.73
[Tan et al. 2018]-Reg.	23.98
[Kim and Suk 2018]	22.77
Ours	**25.63**

Table 2. Results of the user study on recoloring results. We compare our method (Ours) against either the original, ground truth, or previous methods ([Kim and Suk 2018] and [Tan et al. 2018]) using 2AFC pairwise comparisons, and show the percentage of preferred votes by participants on ours under two factors - harmony and naturalness.

vs. Methods	% Preferred Ours	
	Harmony	Naturalness
Original	60.9	65.3
[Kim and Suk 2018]	56.4	65.3
[Tan et al. 2018]-NN	64.9	64.0
[Tan et al. 2018]-Reg.	66.7	64.9
Ground Truth	32.9	52.9

and "Which design looks more natural?" in a two-alternative forced choice (2AFC) manner. The two designs within a pair are two recoloring results from the same photo - one by our method and one original, ground truth or by one of the compared methods ([Kim and Suk 2018] and [Tan et al. 2018]), which were shown side-by-side in randomized order. We take [Tan et al. 2018] as a representative of the palette-based methods in this experiment, as it has a higher PSNR score. All designs were displayed at a resolution of 256 pixels on the short edge. Each HIT consists of 25 different pairs. Each participant is not allowed to work on more than 3 HITs. Each pair is evaluated by more than 3 different participants. We show results in Table 2. It can be seen that our method considerably improves the harmony and naturalness of the original. Though there is still a gap between our results and the ground truth, our method outperforms previous methods in terms of both harmony and naturalness, and achieve similar performance in naturalness to the professional designers. Since [Kim and Suk 2018] is especially designed for recoloring based on a single target color, it has a better performance than the variants of Tan et al. [2018] in harmony.

6.3 Evaluation of the Region Selection Network

During test time, we filter out the region proposals whose confidence scores are less than 0.8, and use non-maximum suppression (NMS) to remove duplicates with IOU larger than 0.4. To generate a binary mask with clear boundary for sending into the recoloring network, we refine the predicted masks with an iterative Conditional Random Field (CRF) used in an existing salient object detection model [Wang et al. 2017]. For fair comparison, we apply the same post-processing step to all the methods mentioned in this section.

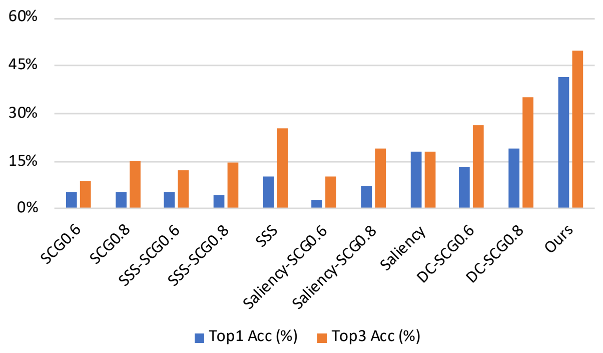


Fig. 12. Region selection performance compared with baselines, image segmentation (**SCG**) [Arbeláez et al. 2014], semantic image segmentation (**SSS**) [Aksoy et al. 2018], saliency detection [Liu et al. 2018], and priors derived form deep colorization (**DC**) [Zhang et al. 2017b].

6.3.1 *Comparison to Baselines.*

To evaluate the quality of the predicted regions, we compare our model with the following baselines on the collected 136 design cases mentioned in Section 6.1.

Image Segmentation: Given an input image, we randomly sample a segment from the results of an image segmentation method (i.e., SCG [Arbeláez et al. 2014]) as the predicted region. We experiment with two different scales, 0.6 (**SCG0.6**) for a finer segmentation and 0.8 (**SCG0.8**) for a coarser, more object-level segmentation. The two scales are chosen since they are close to the empirical mean of sizes of the regions selected by the designers in our testing dataset.

Semantic Segmentation: We first segment an input image with semantic soft segmentation [Aksoy et al. 2018] to separate it into different semantically meaningful regions. We then randomly sample a semantic segment and treat it as the predicted region (**SSS**). As a single semantic region may contain multiple parts with different color distributions in nature, for each semantic segment, we further split it into different sub regions using SCG, and integrate sub regions who have similar mean colors as the predicted region (**SSS-SCG**).

Saliency: We find that human prefer to adjust the color of more salient regions in Section 6.3.4. Inspired by this, we use a state-of-the-art saliency detection method [Liu et al. 2018], to detect a salient region in an input image and treat it as the predicted region (**Saliency**). Further, we combine saliency detection with image segmentation by only sampling image segments obtained by SCG within the salient region (**Saliency-SCG**).

Natural Prior: We use SCG to break an input image into different segments. However, instead of sampling the segments randomly as before, we guide the region prediction using a natural prior learned by a recent deep learning based image colorization method [Zhang et al. 2017b] as baseline (**DC**), which is trained on the same natural image dataset as our model. Given an input grayscale image and a target color, we use its Local Hints Network to predict a probability distribution over the quantized color bins for each pixel and convert it to a continuous distribution using kernel density estimation. The predicted distributions indicate the likely colors for each pixel for image colorization. Therefore, we compute a selection map by using

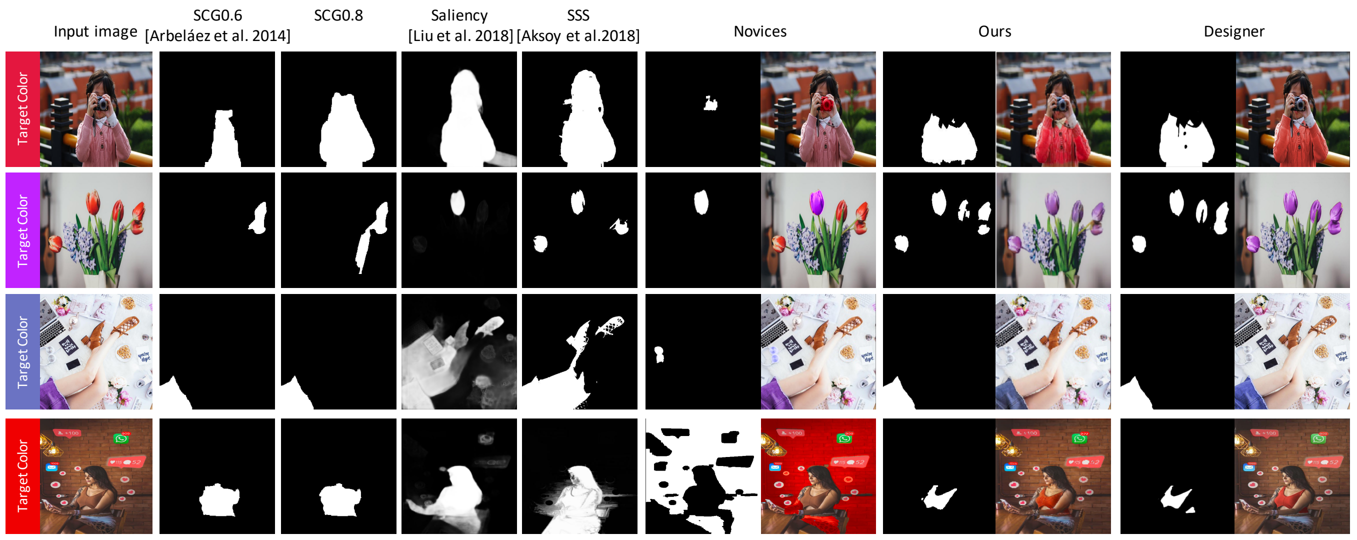


Fig. 13. Comparison of our predicted regions with those generated by baselines, novices and designers. We recolor the regions selected by novices with the pure target color for visual comparison.

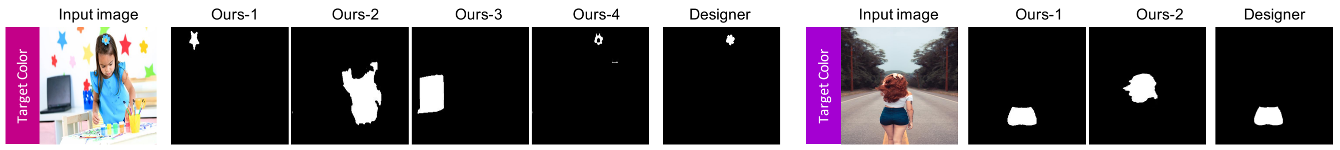


Fig. 14. Results of our region selection model with multiple suggestions.

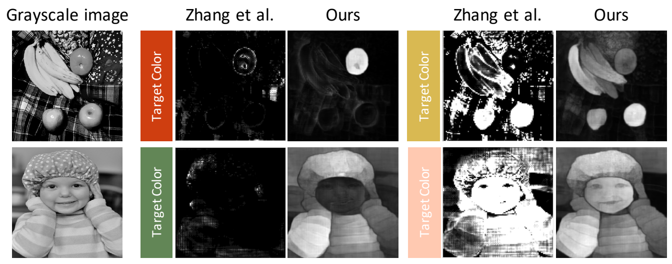
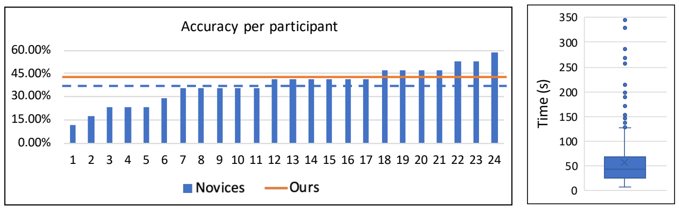


Fig. 15. Selection maps predicted by our network trained on natural images. For comparison, we create a baseline method using the local hints network in deep colorization [Zhang et al. 2017b].

the probabilities of the target color at all the pixels to guide the segment selection.

Results. We show the quantitative results of our method compared with the baselines in Figure 12 and some qualitative comparison results in Figure 13. In view of the generative nature of our task, we report top K accuracy (TopK Acc.), by using each method to predict multiple selected regions and regarding the prediction as correct if any of the top K regions has an intersection-over-union (IoU) of greater than 0.5 with the ground truth. As can be seen in Figure 12, our method outperforms all other methods by a large margin. We analyze the performance of region selection from two aspects: correctness and compactness. For correctness, a random guess with image segmentation (**SCG0.6/0.8**) or semantic segmentation (**SSS**) often gives wrong predictions and has around 5% chance to be correct. By constraining the selection within a salient region (**Saliency**), the accuracy increases slightly. However, a problem with selecting within salient regions is a bias towards choosing foreground regions. While this works in some cases, it will fail miserably when the target colors are more suitable to be assigned to background objects, such as blue for the sky and green for the trees (Figure 17). With the help of the natural prior, we observe large performance improvements, which indicates that the natural prior provides useful information for our task. Our model leads to the best results, demonstrating the advantage of combining natural prior and design knowledge. We also visualize the selection maps of **DC** and ours in Figure 15. As can be seen from the results, our model can localize a more suitable region than theirs based on semantic contents [Zhang et al. 2017b]. It is worth noting that the local hints network serves a different purpose from ours, i.e., finding colors for a given location instead of locations for a given color. To compute a selection map with their approach, we need to evaluate the probability of the target color at each pixel. Their method becomes computationally prohibitive for high-resolution images.

Besides correctness, compactness of the predicted region mask is also one of the main considerations that will have a large impact



(a) (b)

Fig. 16. Results of the user study. (a) Per-user accuracy compared with our average accuracy. The mean accuracy of novices is show as the blue dashed line. (b) Statitsics of the average time that novices spend on each design case, visualized in a box plot.

upon further recoloring. However, we find that it is non-trivial to obtain a compact region with the compared methods. For example, in the first row of Figure 13, both semantic segmentation (**SSS**) and saliency detection (**Saliency**) methods cover the ground truth region well, but tend to generate a larger mask containing multiple regions with different color distributions (i.e., the whole person is selected). For these methods to achieve a compact region selection, a proper segmentation scale is needed, which has to be fine-tuned for different images. If there are several disconnected but related regions that should be changed together, the image segmentation will fail to do so (2^{nd} row of Figure 13). In contrast, our method is able to directly predict a compact local region that is suitable to be changed to the target color.

6.3.2 Comparison to Novices. We conduct a user study to evaluate the performance of our model compared with novices. To this end, we recruited 24 computer science students with no experience on photo recoloring. To allow the participants to efficiently specify a region mask and thus focus on region selection itself, we design an interface that allows them to click on the image segments obtained by SCG for region selection. To do this, for each input image, we pre-calculate a set of image segmentation maps using SCG, and visualize the scale from 0.3 to 0.9 with an interval of 0.1. Users can use a slider bar to switch between different segmentation scales. After a region is selected by a participant, we recolor the selected region with the target color to provide a real-time feedback to him/her.

Each participant was required to complete 17 design cases randomly selected from our test dataset. Each design case was performed by 3 different participants. We only ask them to choose the most suitable region for each design case. We also recorded the time spent by the participants for each design case.

We show accuracy (at a IoU threshold of 0.5) per participant and average accuracy on the test dataset in Figure 16(a). Our method outperforms most of the novices. The variance of accuracies among novices is quite large, and some participants have poor performance. In addition, the novices often need to try several regions before making a decision, taking over 40 seconds on average. In some cases, participants spent more than 3 mins to identify a desirable region. Instead, our model can provide multiple suggestions in a few seconds.

6.3.3 Effect of Learning the Natural Prior. To understand the function of the natural prior in our region selection model, we retrain our model with only S2 and S3, with random initialization, denoted as **Ours (w/o NP)**. We observe that the top1 accuracy drops from 41.6% to 14.6%. We show a qualitative comparison in Figure 17(a). By removing the pre-trained stage on the natural image dataset, the selection maps tend to be blurry without region continuity and may select wrong semantic regions. For example, in the first row, the model gives higher probabilities to skins than tomato given the red color, which is unreasonable. This indicates that pretraining on natural images is necessary for the model to select regions that will result in natural, semantically plausible recoloring and provide a good starting point for further learning design knowledge.

6.3.4 Effect of Learning the Design Knowledge. To validate the importance of training with graphic design data, we retrain the model by skipping the S2 and only train S3 on top of S1 to see whether the selection map and region mask accuracy benefit from fine-tuning with graphic design data (**Ours w/o DK**). After removing fine-tuning on the selection map, the accuracy decreases to only 5%. To find out the reasons for this huge performance drop, we visualize the selection maps before and after fine-tuning on graphic design data in Figure 17(b). From the results (1^{st} and 3^{rd} row) we can see that, with design knowledge, the model tries to emphasize more on salient regions when there are multiple equal probability regions (in **Ours w/o DK**). This conforms with our common sense that people are more sensitive to salient regions and adjusting the color of these salient regions can magnify the color effects. Besides, the selection map becomes more discriminative after involving design knowledge. In this way, the region proposal sub-net is easier to find the regions designers prefer, thereby increasing the accuracy on the test dataset.

6.3.5 Effect of Mask Sub-net. We retrain with all 3 stages but remove the mask sub-net from S2 to see its advantage, denoted as **Ours (w/o mask)**. All other settings remain the same as in Section 4.4. To calculate the accuracy after removing the mask sub-net in **Ours (w/o mask)**, we directly use the proposal bounding box to crop the selection map as the predicted mask. Without the mask sub-net, top-1 accuracy drops from 41.6% to 22.6%, indicating the importance of the mask sub-net. After visualizing the selection maps in Figure 17(c), we find that the mask sub-net also helps refine the selection map by localizing more compact and well-aligned regions. This is because the mask sub-net and selection map sub-net share the same feature maps, and the mask sub-net forces the feature maps to encode more low-level details for predicting a more suitable mask, which in turn benefits the selection map prediction.

6.4 Evaluation of Recoloring Network

We compare our recoloring network with two kinds of baselines: a traditional color propagation [Levin et al. 2004], and a deep colorization method [Zhang et al. 2017b]. For the stroke-based method, we use the target color to draw strokes within the selected region and original colors for the other regions. The results are shown in Figure 18. With only a single color, the colors within the mask region are flat and less vivid, although we have manually added a lot of strokes.

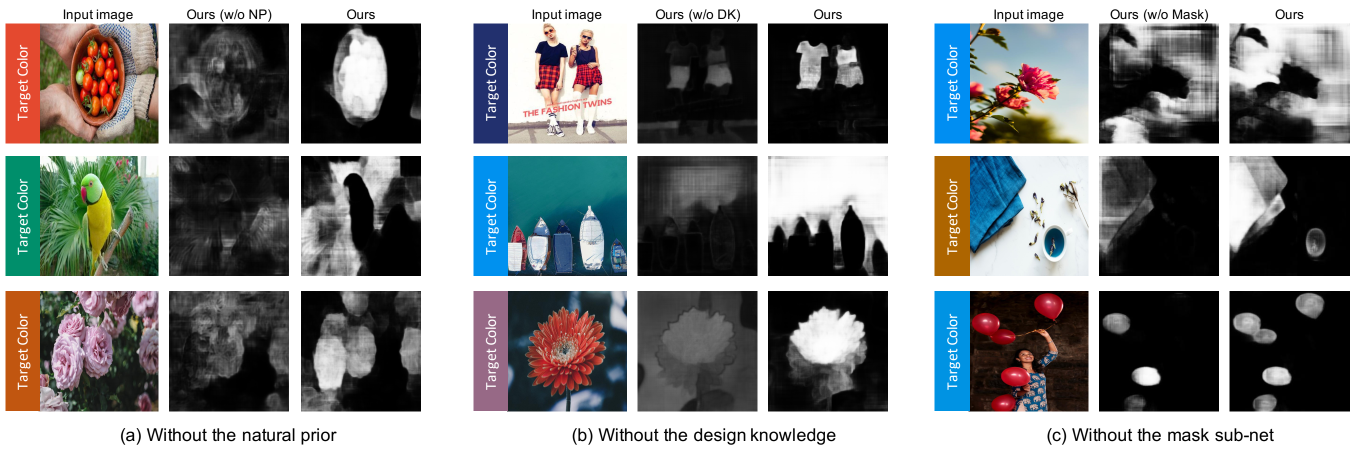


Fig. 17. What does our region selection network learn? We show selection maps predicted by variants of our model. (a) Training with the natural prior helps the model locate semantically meaningful regions for the target color, and provides a good starting point for further learning design knowledge. (b) Training with the design knowledge helps the model generate more discriminative selection maps, reducing the ambiguities in region selection. (c) Training with the mask sub-net is helpful to predict more compact and well-aligned regions.

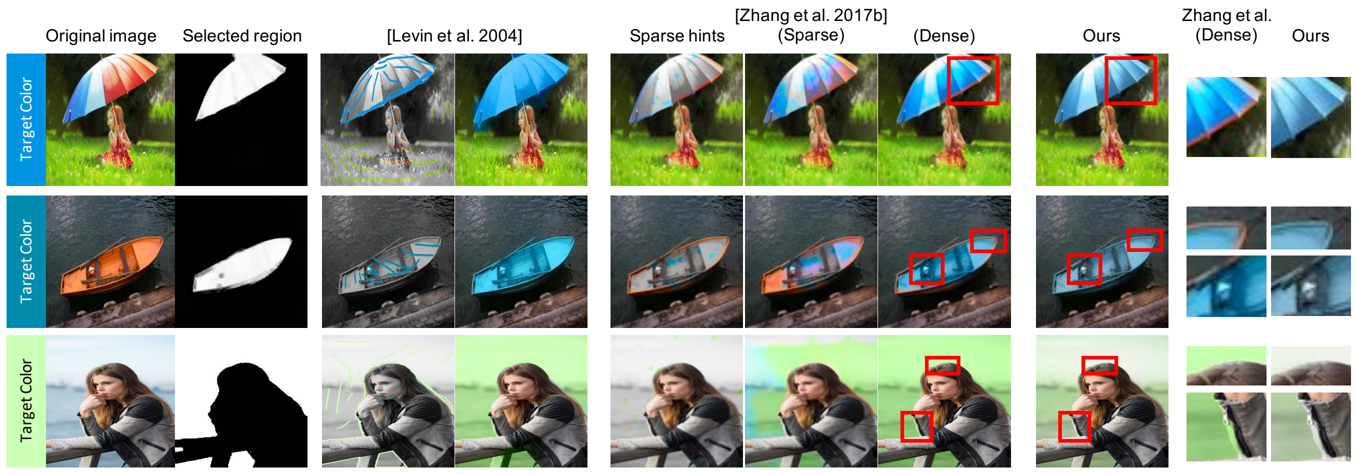


Fig. 18. Comparison between traditional stoke-based color propagation method [Levin et al. 2004], deep colorization [Zhang et al. 2017b] and our recoloring network. To adapt deep colorization model to the recoloring task, we copy the pixels of the original images outside the mask as hint points. In Sparse mode, we randomly sample 20 hint points colored as the target color within the mask. In Dense mode, we treat each pixel within the mask as hint points colored by the target color. Our recoloring network propagates the target color more naturally, especially near the boundary. We show closeups of the results generated by Zhang *et al.* [2017b] (the Dense mode) and our method on the right.

For the deep colorization model, to adapt to our recoloring task, we use the original color as hints for non-selected regions. We use the target color as a color hint on each pixel inside the selected region in the *Dense* mode, and randomly select 20 points in the *Sparse* mode. Figure 18 shows the results. In the Sparse mode, the hint points need to be manually given and carefully distributed within the selected region. Otherwise, the target color may not be propagated uniformly in the selected region. When the hint points are given in the Dense mode, the model may directly copy the target color into each pixel without much modification. Both modes cannot propagate the target color correctly near the mask boundary. With the help of transition region in the recoloring network, our method can naturally propagate the color near the mask boundary without generating artifacts. Note that all the baselines require more or less human inputs and suffer from artifacts near the region boundary, while our recoloring model can achieve natural results automatically.

We also compare our recoloring results with those generated by designers qualitatively in Figure 13. Both the designers and our method can propagate target colors naturally in the selected regions. Note that our results are very close to what the designers produce, although some subtle color differences can be observed. Since the

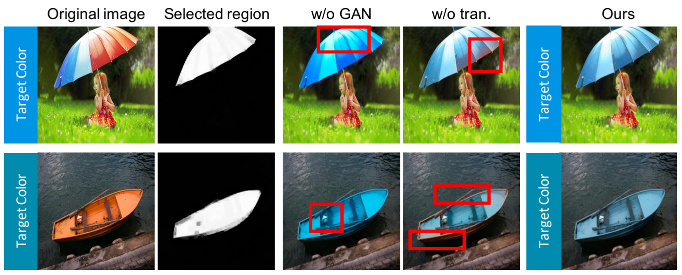


Fig. 19. Ablation studies on our recoloring network.

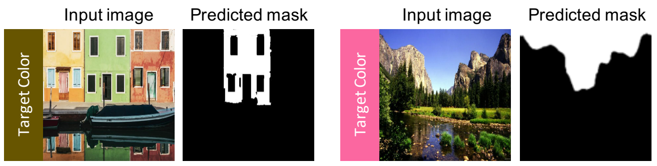


Fig. 20. Failure case. Without considering global consistency, the related regions (buildings/sky and their reflections) may not be selected together.

perception of color naturalness varies across different people, the designers and our model (learned from data by different designers) may tweak the target colors in slightly different ways to make the entire images look natural, which causes the differences.

To understand the roles of the GAN loss and the boundary region. We compare our full network with its two variants: one trained without the GAN loss in Eq. 8 (**w/o GAN**), one without dropping out transition regions in the input color (**w/o tran.**). We show the results in Figure 19. After removing the GAN loss, the model tries to directly copy the target color to the selected region with less variation on different textures, which makes the results look very offensive and unnatural. After removing the transition region, the colors fail to propagate naturally near the boundary. With the help of the GAN loss and transition region, our recoloring model learns to fill the selection regions with more realistic colors that are coherent with the contexts.

7 CONCLUSIONS AND FUTURE WORK

In this paper, we propose a selective region-based photo recoloring approach for graphic designs, which can automatically and naturally adjust the colors of input photos to match the target colors from the graphic designs. To address the main challenge of selecting appropriate regions to be recolored to the target colors, we propose a region selection model, which is trained on both natural images and graphic designs. This allows our model to capture both natural prior and design heuristics, and apply them to effectively predict suitable regions for recoloring at test time. Extensive qualitative and quantitative evaluations demonstrate the superiority of our method over existing photo recoloring works and various baselines.

Our method still has limitations. First, our model is designed for finding local modifiable regions without considering global consistency among different semantic regions. Therefore, related regions and objects cross different semantics in an input photo may not be selected together, and thus will be recolored in different ways. As shown in Figure 20, the reflections are not selected along with the objects that cause the reflections (i.e., the building and sky). We consider solving this problem in the future by enforcing some non-local constraints between regions with different semantics. Second, our model assumes grayscale images as input. While this makes it possible for us to learn from synthetic data, it losses color context information that may be important for region selection. In practice, people sometimes take into account global color harmonization and local interaction between a color region and its context to determine what region to modify. To address it, jointly leveraging our model with existing color harmonization model [Cohen-Or et al. 2006] would be an interesting direction to explore further. Third, when using multiple target colors, more than one color may be mapped to the same region, such as the balloon in the last example of Figure 1. Our current workaround is to simply assign the region to the color with the highest mask confidence score. We will leave more advanced methods to handle this case as future work. Lastly, as a data-driven method, the size of the graphic design dataset may restrict the ability of our region selection network, and the network may prefer to select regions that have appeared in the dataset. We believe that a larger dataset can alleviate this problem, and is possible to be obtained with the growth of online graphic design communities.

ACKNOWLEDGMENTS

We thank the anonymous reviewers for their valuable feedback. The work described in this paper was partially supported by a GRF grant from the Research Grants Council of Hong Kong (RGC Ref. No.: 11205620).

REFERENCES

Yağız Aksoy, Tunç Ozan Aydın, Aljoša Smolić, and Marc Pollefeys. 2017. Unmixing-Based Soft Color Segmentation for Image Manipulation. *ACM TOG* 36, 2 (2017), 19:1–19:19.
Yağiz Aksoy, Tae-Hyun Oh, Sylvain Paris, Marc Pollefeys, and Wojciech Matusik. 2018. Semantic soft segmentation. *ACM TOG* 37, 4 (2018), 72.
P. Arbeláez, J. Pont-Tuset, J. Barron, F. Marques, and J. Malik. 2014. Multiscale Combinatorial Grouping. In *Proc. IEEE CVPR*.
Huiwen Chang, Ohad Fried, Yiming Liu, Stephen DiVerdi, and Adam Finkelstein. 2015. Palette-based photo recoloring. *ACM TOG* 34, 4 (2015), 139.
Youngha Chang, Suguru Saito, Keiji Uchikawa, and Masayuki Nakajima. 2005. Example-based color stylization of images. *ACM TAP* 2, 3 (2005), 322–345.
Daniel Cohen-Or, Olga Sorkine, Ran Gal, Tommer Leyvand, and Ying-Qing Xu. 2006. Color harmonization. In *ACM TOG*, Vol. 25. 624–630.
Yuki Endo, Satoshi Iizuka, Yoshihiro Kanamori, and Jun Mitani. 2016. Deepprop: Extracting deep features from a single image for edit propagation. In *CGF*, Vol. 35. Wiley Online Library, 189–201.
Leon A Gatys, Alexander S Ecker, and Matthias Bethge. 2016. Image style transfer using convolutional neural networks. In *Proc. IEEE CVPR*. 2414–2423.
Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. In *Proc. NIPS*. 2672–2680.
Yoav HaCohen, Eli Shechtman, Dan B Goldman, and Dani Lischinski. 2013. Optimizing color consistency in photo collections. *ACM TOG* 32, 4 (2013), 38.
Mingming He, Dongdong Chen, Jing Liao, Pedro V Sander, and Lu Yuan. 2018. Deep exemplar-based colorization. *ACM TOG* 37, 4 (2018), 47.
H-Z Huang, S-H Zhang, RR Martin, and S-M Hu. 2014. Learning natural colors for image recoloring. In *CGF*, Vol. 33. Wiley Online Library, 299–308.
Satoshi Iizuka, Edgar Simo-Serra, and Hiroshi Ishikawa. 2017. Globally and locally consistent image completion. *ACM TOG* 36, 4 (2017), 107.
Sergey Ioffe and Christian Szegedy. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv:1502.03167* (2015).

Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. 2017. Image-to-image translation with conditional adversarial networks. *Prof. IEEE CVPR* (2017).

EunJin Kim and Hyeon-Jeong Suk. 2018. Image color adjustment for harmony with a target color. *Color Research & Application* 43, 1 (2018), 75–88.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv:1412.6980* (2014).

Anat Levin, Dani Lischinski, and Yair Weiss. 2004. Colorization using optimization. In *ACM TOG*, Vol. 23. 689–694.

Yuanzhen Li, Edward Adelson, and Aseem Agarwala. 2008. ScribbleBoost: Adding Classification to Edge-Aware Interpolation of Local Image and Video Adjustments. In *CGF*, Vol. 27. Wiley Online Library, 1255–1264.

Jing Liao, Yuan Yao, Lu Yuan, Gang Hua, and Sing Bing Kang. 2017. Visual attribute transfer through deep image analogy. *ACM TOG* 36, 4 (2017), 120.

Nian Liu, Junwei Han, and Ming-Hsuan Yang. 2018. PiCANet: Learning pixel-wise contextual attention for saliency detection. In *Proc. IEEE CVPR*. 3089–3098.

Fujun Luan, Sylvain Paris, Eli Shechtman, and Kavita Bala. 2017. Deep photo style transfer. *CoRR, abs/1703.07511* 2 (2017).

Qing Luan, Fang Wen, Daniel Cohen-Or, Lin Liang, Ying-Qing Xu, and Heung-Yeung Shum. 2007. Natural image colorization. In *Proc. EG Rendering Techniques*. Eurographics Association, 309–320.

Xudong Mao, Qing Li, Haoran Xie, Raymond YK Lau, Zhen Wang, and Stephen Paul Smolley. 2017. Least squares generative adversarial networks. In *Proc. ICCV*. IEEE, 2813–2821.

Tiziano Portenier, Qiyang Hu, Attila Szabo, Siavash Bigdeli, Paolo Favaro, and Matthias Zwicker. 2018. FaceShop: Deep Sketch-based Face Image Editing. *arXiv:1804.08972* (2018).

Joseph Redmon and Ali Farhadi. 2018. Yolov3: An incremental improvement. *arXiv:1804.02767* (2018).

Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. 2015. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Proc. NIPS*. 91–99.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. 2015. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*. Springer, 234–241.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. 2015. Imagenet large scale visual recognition challenge. *Proc. ICCV* 115, 3 (2015), 211–252.

Maja Konecnik Ruzzier and Leslie De Chernatony. 2013. Developing and applying a place brand identity model: The case of Slovenia. *Journal of Business Research* 66, 1 (2013), 45–52.

Jianchao Tan, Jose Echevarria, and Yotam Gingold. 2018. Efficient palette-based decomposition and recoloring of images via RGBXY-space geometry. In *ACM TOG*. 262.

Robert L Underwood. 2003. The communicative power of product packaging: creating brand identity via lived and mediated experience. *Journal of Marketing Theory and Practice* 11, 1 (2003), 62–76.

Orana Velarde. 2018. Quick Tips for Creating Social Media Graphics Like a Designer. https://visme.co/blog/add-text-to-image/.

Baoyuan Wang, Yizhou Yu, Tien-Tsin Wong, Chun Chen, and Ying-Qing Xu. 2010. Data-driven image color theme enhancement. *ACM TOG* 29, 6 (2010), 1–10.

Lijun Wang, Huchuan Lu, Yifan Wang, Mengyang Feng, Dong Wang, Baocai Yin, and Xiang Ruan. 2017. Learning to detect salient objects with image-level supervision. In *Proc. IEEE CVPR*. 136–145.

Liron Yatziv and Guillermo Sapiro. 2006. Fast image and video colorization using chrominance blending. *IEEE TIP* 15, 5 (2006), 1120–1129.

Lvmin Zhang, Chengze Li, Tien-Tsin Wong, Yi Ji, and Chunping Liu. 2018. Two-stage sketch colorization. In *ACM TOG*. 261.

Qing Zhang, Chunxia Xiao, Hanqiu Sun, and Feng Tang. 2017a. Palette-based image recoloring using color decomposition optimization. *IEEE TIP* 26, 4 (2017), 1952–1964.

Richard Zhang, Jun-Yan Zhu, Phillip Isola, Xinyang Geng, Angela S Lin, Tianhe Yu, and Alexei A Efros. 2017b. Real-time user-guided image colorization with learned deep priors. *ACM TOG* (2017).

A THE COLLECTED GRAPHIC DESIGN DATASET

Here are some statistics of our newly collected graphic design dataset, including distribution of region number per image (Figure 21), target colors (Figure 22), region size (Figure 23), and region location (Figure 24).

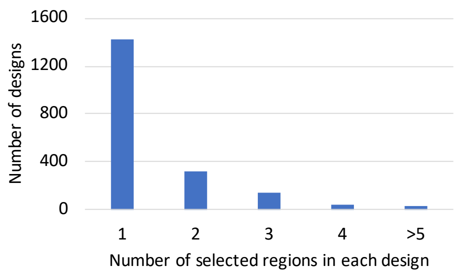


Fig. 21. The distribution of the number of selected regions per image in our collected graphic design dataset.

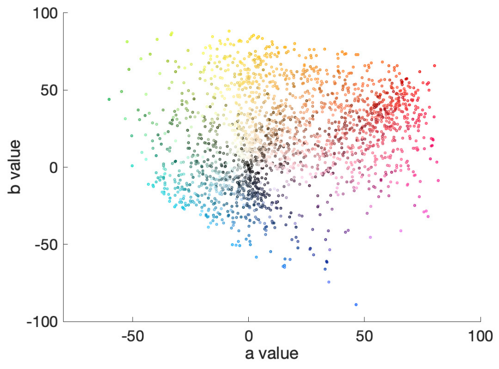


Fig. 22. The distribution of target colors, represented as ab values in the CIE Lab space.

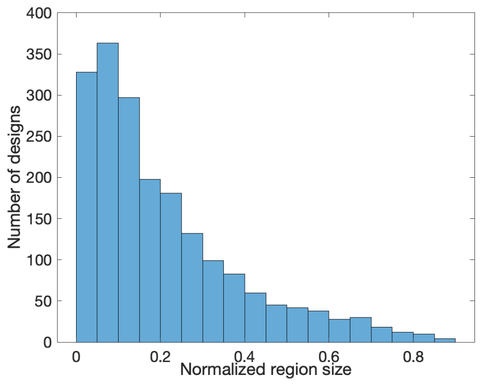


Fig. 23. The distribution of region sizes.

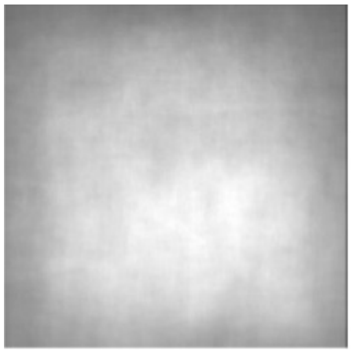


Fig. 24. Spatial distribution of selected regions. The intensity map is obtained by rescaling all the designs into squares of the same size and then averaging the scaled binary segmentation masks of all the selected regions. It specified the likelihood a region appear at each location, with higher intensities indicating higher likelihood.