Language-based Colorization of Scene Sketches

CHANGQING ZOU*, Sun Yat-sen University and Huawei Noah's Ark Lab
HAORAN MO*, Sun Yat-sen University
CHENGYING GAO†, Sun Yat-sen University
RUOFEI DU‡, Google
HONGBO FU, City University of Hong Kong

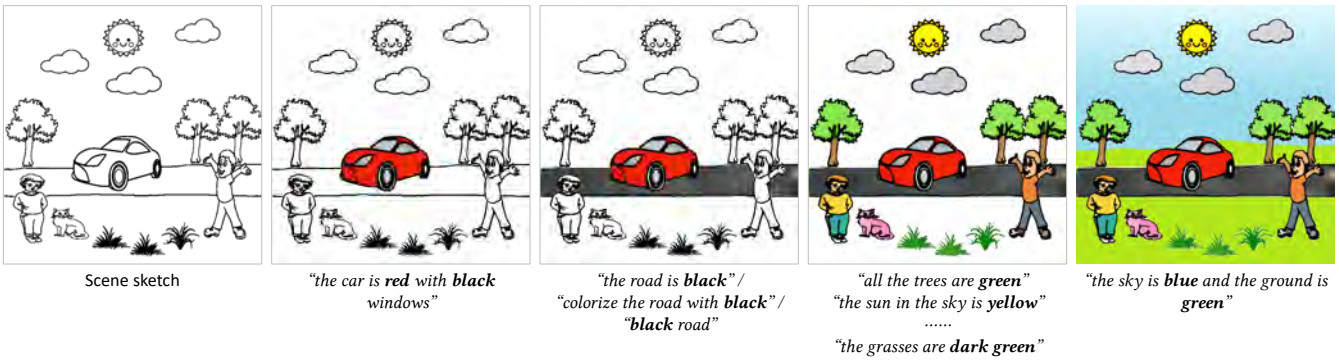

Fig. 1. Given a scene sketch, our system automatically produces a colorized cartoon image by progressively coloring foreground object instances and the background following user-specified language-based instructions.

Being natural, touchless, and fun-embracing, language-based inputs have been demonstrated effective for various tasks from image generation to literacy education for children. This paper for the first time presents a language-based system for interactive colorization of scene sketches, based on semantic comprehension. The proposed system is built upon deep neural networks trained on a large-scale repository of scene sketches and cartoon-style color images with text descriptions. Given a scene sketch, our system allows users, via language-based instructions, to interactively localize and colorize specific foreground object instances to meet various colorization requirements in a progressive way. We demonstrate the effectiveness of our approach via comprehensive experimental results including alternative studies, comparison with the state-of-the-art methods, and generalization user studies. Given the unique characteristics of language-based inputs, we envision a combination of our interface with a traditional scribble-based interface for a practical multimodal colorization system, benefiting various applications. The dataset and source code can be found at https://github.com/SketchyScene/SketchySceneColorization.

CCS Concepts: • **Computing methodologies** → **Image Processing**;

Additional Key Words and Phrases: Deep Neural Networks; Image Segmentation; Language-based Editing; Scene Sketch; Sketch Colorization

*Both authors contributed equally to the paper.
†Corresponding author: mcsgcy@mail.sysu.edu.cn
‡This project was started before this author joined Google.

0730-0301/2019/11-ART233 $15.00
https://doi.org/10.1145/3355089.3356561

ACM Reference format:
Changqing Zou, Haoran Mo, Chengying Gao, Ruofei Du, and Hongbo Fu. 2019. Language-based Colorization of Scene Sketches. *ACM Trans. Graph.* 38, 6, Article 233 (November 2019), 16 pages.
https://doi.org/10.1145/3355089.3356561

1 INTRODUCTION

In recent years, deep learning techniques have significantly improved the performance of natural language processing [Kim 2014; Lai et al. 2015; Lample et al. 2016]. Smart speakers such as *Google Home* and *Amazon Alexa* are widely used and offer hands-free interactions through language-based instructions. This has motivated researchers to explore language-based instructions as an alternative input to crucial problems such as image editing and generation [Cheng et al. 2014; Laput et al. 2013; Park et al. 2019; Xu et al. 2018; Yan et al. 2016; Zhang et al. 2017a], and object retrieval [Hu et al. 2016b; Li et al. 2017].

Despite the fact that a language-based interface might not provide fine or direct control of results, it does provide unique characteristics including being natural to everyone and being touchless, allowing for interesting applications that are difficult or even impossible to achieve with existing interfaces. Recent studies have also found its unique strength for children. For example, Lovato and Piper [2015] found that voice input is mainly used for exploration and fun by children. A recent study in [Jung et al. 2019] showed that voice-based interaction leads children to be more immersed in an educational programming game. Moreover, Raffle et al. [2007] also found that embedding sound and voice in traditional drawings by

recording children's storytelling while painting could provide a significantly more effective system to support the literacy development of children.

In this work, we present a language-based system for interactive colorization of scene sketches, with respect to text-based color specifications (Fig. 1). Although traditional interactive sketch colorization solutions (e.g., [Qu et al. 2006; Sangkloy et al. 2017]) support precise control via a scribble-based interface, they require explicit color selection using color picking tools and direct target selection. By contrast, when text-based descriptions are given through voice input, our system can be more easily adopted by novice users, and is friendly for people with upper limb impairments. Our system can potentially aid in the cognitive development of children through tasks such as color and object recognition. In addition, indirect association between language-based instructions and scene sketches also allows easy reusing of the same set of instructions as a theme for consistently colorizing multiple different sketches involving a similar set of objects. This is challenging for scribble-based colorization interfaces due to their direct and fixed relationship between scribbles and specific sketch regions. We thus believe that a language-based interface is complementary to a scribble-based interface for colorization tasks, and envision a practical multimodal colorization system that takes advantages of both types of interfaces.

Our task is challenging mainly because of the unknown and indirect mapping between the input language-based instructions and the scene sketches. More specifically, the first challenge is how to automatically localize and segment target objects indicated by the language-based instructions. High-quality segmentation results are crucial to the subsequent colorization process, as users are likely to specify different colors for individual objects. This problem of text-based instance segmentation for scene sketches has not been explored before. To address this issue, we propose a new architecture, called *instance matching model*, which integrates sophisticated networks for sketch feature extraction and multimodal (textual, visual, and spatial information) feature fusion, together with a training strategy tailored for scene sketches upon a text-based instance segmentation dataset consisting of 38K triplet samples of scene sketches, text descriptions, and instances.

The second challenge is how to colorize an individual target object instance with respect to language-based inputs. This challenge requires our system to automatically build accurate correspondence between object instances and text-based color specifications (e.g., in Fig. 1, given the expression *"the car is red with black windows"*, black, rather than red, should be assigned to the car windows). Additionally, a user might wish to assign different colorization goals to different object parts, e.g., colorizing the car body and car windows in Fig. 1. It therefore requires the system to learn object-part-level segmentation and colorization from natural language expressions. To tackle these challenges, we embedded the text-image interaction model mLSTM (multimodal Long Short-Term Memory) [Liu et al. 2017a] to a Generative Adversarial Network (GAN) to perform language-based colorization. To train this network, we collected a large-scale dataset consisting of 4K triplet samples of object sketches, text descriptions, and colorized instances, as well as 20K quadruple samples of color foregrounds, text descriptions, colorized backgrounds, and segmentation label maps.

Our system automatically responds to user-specified colorization instructions and colorizes target objects in an input scene sketch. Our system allows a certain range of expression structure variances and language grammar mistakes for user-specified instructions. For example, given different expressions but with the same intention like *"the road is black"*, *"colorize the road with black"*, and *"black road"*, our system leads to the same colorization results (Fig. 1). Our system is also able to deal with one or multiple object instances of the same category with a single instruction. For example, given an instruction *"all the trees are green"* in Fig. 1, our system colorizes all tree instances in green. Our experimental results in Section 8 show that the proposed colorization system achieves visually pleasing results, as confirmed by multiple user studies. The impact of individual components is validated by a set of ablation studies.

We highlight our main contributions as follows:

(1) The first language-based user-customizable colorization system for scene sketches.
(2) The first architecture for text-based instance-level segmentation of scene sketches.
(3) Three large-scale datasets for text-based instance segmentation, foreground colorization, and background colorization.

2 RELATED WORK

2.1 Language-based Image Segmentation

Image segmentation guided by natural language expressions has attracted increasing attention recently, due to the advance of semantic image segmentation and natural language processing. Hu *et al.* [2016a] proposed the first language-based (referring) image segmentation technique, which directly outputs a binary segmentation mask of a single target object given a natural language description as a query. Their technique was improved by Liu *et al.* [2017a] by employing multimodal feature fusion with a recurrent multimodal interaction (RMI) model, which encodes sequential interactions between textual, visual, and spatial information. Li *et al.* [2018] proposed a refinement network to improve Hu *et al.*'s work by feeding late-fused multimodal features back to low-level layers of an image encoder with a convolutional LSTM to incorporate multi-scale semantics for better segmentation results. Shi *et al.* [2018] argued that extracting key words would be helpful to suppress the noise in the query and to highlight desired objects. Following the basic framework of RMI, they proposed a key-word-aware network, which contains a query attention model and a key-word-aware visual context model for referring image segmentation. Apart from these works, there have also been other studies focusing on visual grounding, which aim to locate the most relevant object or region in an image by a bounding box [Hu et al. 2016b; Mao et al. 2016] or an attention region [Lu et al. 2016; Yu et al. 2017] based on a natural language query.

Our instance matching model is closely related to the above approaches. However, our matching model takes as input natural language expressions and a scene sketch, rather than a natural image. In addition, our matching model aims to infer the segmentation masks of one or multiple object instances of interest, including the information of bounding box, binary instance mask, and class label. Therefore, the output of our matching model is different from the

existing works in referring image segmentation or visual grounding, which output the binary segmentation or bounding box of a single target object instance.

2.2 User-customized Image Colorization

This task generates color images from gray-scale or sketch images based on user inputs. Several types of user inputs currently exist, including user-drawn color scribbles, user-chosen color reference images, user-selected palettes, and user-specified language expressions. A scribble-based interface has been commonly adopted to specify desired colors on a gray-scale or sketch image [Ci et al. 2018; Liu et al. 2017b; Qu et al. 2006; Sangkloy et al. 2017; Zhang et al. 2018, 2017b]. There have been several open-sourced interactive applications for scribble-based line-drawing colorization, such as *PaintsChainer* [Yonetsuji 2017] and *Style2paints* [Zhang et al. 2018]. As discussed previously, the scribble-based and language-based interfaces have their own advantages, the former for being direct and supporting precise control while the latter for being more natural and accessible.

The reference image based colorization task [Fang et al. 2019; Furusawa et al. 2017; He et al. 2018; Wang et al. 2017] takes color reference images as input, and achieves colorization by properly transferring colors from reference images to gray-scale or sketch images. Such methods greatly reduce the degree of user intervention (when colorization results are visually plausible) but at the same time do not allow flexible control of colorization results. The palette-based colorization approach [Chang et al. 2015] allows users to specify a certain number of colors (up to 5 colors) from a palette to control colorization results. This line of work mainly focuses on automatic global color style transfer, and hence it also does not support a high degree of user customization. Some recent studies have been exploring multiple modalities (e.g., color palettes and scribbles [Xiao et al. 2019b], reference images and palettes [Xiao et al. 2019a]) as input for user-customized colorization. The multiple input modalities of the existing approaches do not contain the modality of natural language expression.

The concept of language-based colorization was first introduced by Chen *et al.* [2018b] in their language-based image editing (LBIE) framework. Their framework contains a recurrent model using attention mechanism for feature fusion between a natural language expression and an image, and thus allows language-based colorization for object-level edge maps or gray-scale images. Bahng *et al.* [2018] addressed the problem of image colorization with color hints implicitly given by an input text (e.g., coloring a gray-scale image of bird based on a phrase like *"rose sensations of sky"*). Their solution focuses on the generation of color palettes to reflect the semantics of an input text.

Our system is most close to LBIE as it is the first natural language based colorization framework for scene sketches. Directly using LBIE for our problem is infeasible because of the lack of pair-wise scene sketch and color image data. We address this challenge by decomposing a scene sketch into foreground object instances and background regions, and make this problem solvable by training deep networks on pair-wise object-level sketch and color image data that are easy to collect from the Internet or existing datasets. Although LBIE can be directly used for both foreground object instance colorization and background colorization, the experiments in Section 8 show that it is less effective than our networks. There are some other works that study the mutual reasoning and inference between language expressions and colors [Monroe et al. 2016, 2017]. These works could potentially be combined with our system to enable more accurate color control.

3 SYSTEM OVERVIEW

Our current system takes text-based colorization instructions as input. Such instructions can be obtained through voice or typing. The voice-based interface is more natural but might suffer from speech-to-text errors. Integrating our system with direct voice input would be interesting future work.

As illustrated in Fig. 2, given an input scene sketch and a natural language expression for color specification, our system offers two modes for colorization: foreground and background. Two modes are adopted because foreground objects (e.g., cars, trees, sun) and background regions (e.g., sky, ground) have very different image characteristics, and thus are better dealt with in different ways. In our system, we classify all sketched objects as foreground and the regions between sketched objects as background. Assuming a user does not colorize foreground objects and background regions in a single language expression, either the foreground or background mode can be easily determined by checking the category label in a given instruction (e.g., "sky" indicates the background label.)

In the foreground mode, a network called *instance matching model* (Section 4) is first used to locate the foreground object instance(s) of interest indicated by the natural language instruction (more precisely, predicting the instance-level mask of the target object instance(s)). Next, a new network architecture called *foreground colorization model* (Section 5.1) specifically designed for foreground objects is employed to colorize these instance(s). In the background mode, a third network architecture called *background colorization model* (Section 5.2) specifically designed for background stuff is employed to perform simultaneous segmentation and colorization. We do not use a specific instance matching model for the background, since the colorization requirements for background regions are less complex than those of foreground object instances. With this divide-and-conquer and progressive strategy, the colorization of a complex scene sketch becomes feasible, without being trained on a large-scale set of scene-level sketch and image pairs with the entire annotated text instructions.

4 INSTANCE MATCHING

The instance matching model takes as input a scene sketch image and a language-based instruction (a phrase or sentence), and outputs the pixel-level mask of the target object instance(s), including the information of bounding box, class label, and binary instance mask. This problem is challenging and there exists almost no prior work directly studying it. We refer to this problem as *referring instance segmentation* and address it with a new architecture integrating a set of sophisticated networks.

Our proposed architecture for instance matching as illustrated in Fig. 3 mainly includes two phases: sketch image feature extraction,

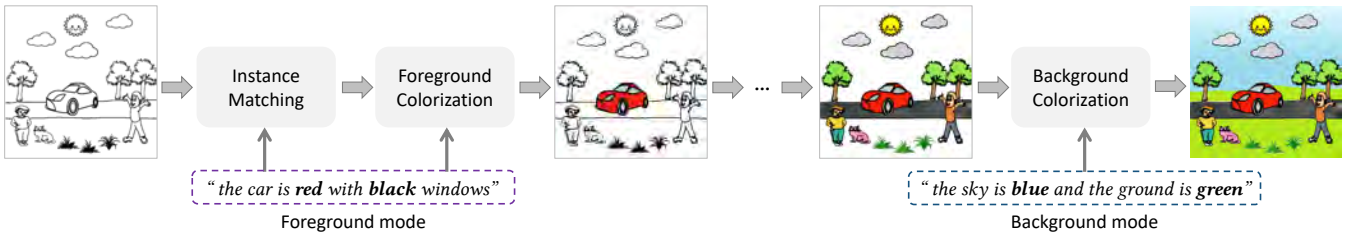

Fig. 2. Our system supports two-mode interactive colorization for a given input scene sketch and text-based colorization instructions, using three models, namely, the instance matching model, foreground colorization model, and background colorization model. It is not necessary to colorize foreground objects before background regions.

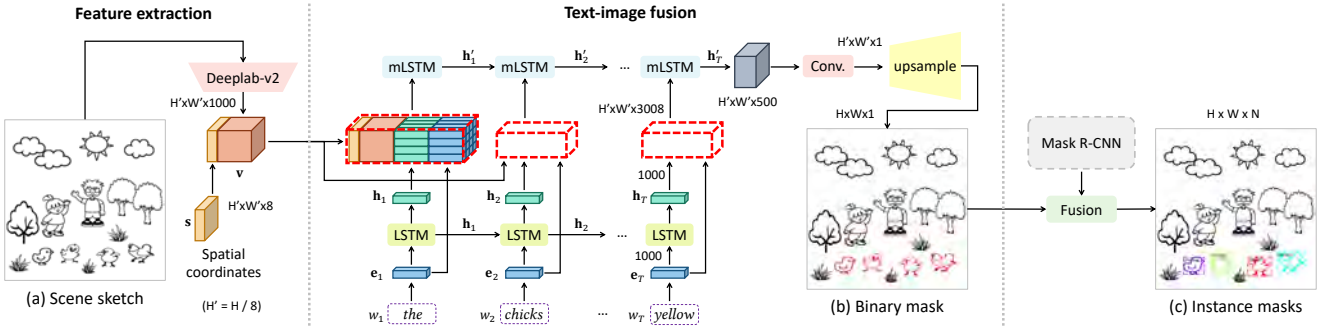

Fig. 3. Network architecture of the instance matching model in Section 4. This network is trained in an end-to-end manner to obtain the binary mask (shown in (b)). In the inferring phase, the generated binary mask is fused with the instance segmentation results generated by Mask R-CNN [He et al. 2017] to obtain the final results.

and text-image fusion. The former one extracts the image features of the sketch, and the latter takes them along with a natural language description as inputs, and generates the binary mask of the target object(s) (Fig. 3 (b)). We use the DeepLab-v2 network [Chen et al. 2018a] as the sketch image feature extractor as it is the most effective network for semantic segmentation of scene sketches according to the study in [Zou et al. 2018]. The RMI model [Liu et al. 2017a] which was originally proposed for referring image segmentation, is employed for text-image fusion phase. The final instance-level segmentation information is obtained by fusing the binary mask(s) and the results generated by Mask R-CNN [He et al. 2017] (separately trained). The *ignoring-background* training strategy [Zou et al. 2018] tailored for sketch data, which only penalizes the cross entropy loss of stroke pixels rather than every pixel in a sketch image, is leveraged to train all the networks. In the remainder of this paper, we refer to this architecture empowered by DeepLab-v2 and RMI as well as the *ignoring background* training strategy as **DeepLabv2-RMI**.

Methodology. In the sketch image feature extraction phase, given an image of size $H \times W$, a ResNet-101 based DeepLab-v2 model is adopted to extract sketch image features with size $H' \times W' \times 1000$ where $H' = H/8$ and $W' = W/8$. The sketch image features are then concatenated with spatial coordinates to produce a $H' \times W' \times (1000 + 8)$ tensor. The 8 spatial coordinate dimensions, where the normalized horizontal and vertical positions individually use 3 dimensions each and the remaining 2 dimensions are $1/W'$ and $1/H'$, are determined by following the implementation of [Liu et al. 2017a].

In the text-image fusion phase, a two-layer LSTM architecture is employed for multimodal interaction between cross-domain features. The text-only LSTM encodes the language instruction consisting of several words $\{w_t\}_{t=1}^{T}$ from their mapping embedding $\{\mathbf{e}_t\}_{t=1}^{T}$ ($\mathbf{e}_t \in \mathbb{R}^e$) and uses the hidden representation $\{\mathbf{h}_t\}_{t=1}^{T}$ ($\mathbf{h}_t \in \mathbb{R}^w$) for each word as the text features. At each time step t, the four kinds of cross-domain features (text embedding $\mathbf{e}_t$, text semantic $\mathbf{h}_t$, extracted image features $\mathbf{v}$ and spatial information $\mathbf{s} \in \mathbb{R}^s$) are concatenated as a joint input for the convolutional multimodal LSTM (mLSTM). Before concatenation, $\mathbf{e}_t$ and $\mathbf{h}_t$ are initially tiled to $H' \times W'$ to match the dimensions. The mLSTM is applied to all spatial locations in the concatenated feature maps, and output hidden states $\{\mathbf{h'}_t\}_{t=1}^{T}$ ($\mathbf{h'}_t \in \mathbb{R}^m$). The binary mask $R \in \mathbb{R}^{W \times H}$ is produced from $\mathbf{h}'_T$ via a projection and an upsampling layer. The network for binary mask generation is trained in an end-to-end manner. On the inferring stage, the generated binary mask is fused with the output of Mask R-CNN to produce final instance segmentation results. Specifically, the segmented instances from Mask R-CNN with more than 50% mask pixels covered by the generated binary mask are used as the matched instances, i.e., the final segmentation results.

Training Loss. Given a binary drawing mask of a sketch $M \in \mathbb{R}^{W \times H}$ (where $M^{ij} = 1$ indicates a black foreground pixel at position (i, j) and $M^{ij} = 0$ indicates a white background pixel) and the ground truth binary segmentation mask $\hat{R}$, the loss function is formulated as the conventional cross-entropy empowered by the

ignoring background training strategy:

$$L = \frac{1}{N}\sum_{i=1}^{W}\sum_{j=1}^{H}\left(M^{ij} * \left(\hat{R}^{ij} * -\log\left(R^{ij}\right) + (1-\hat{R}^{ij}) * -\log\left(1-R^{ij}\right)\right)\right) \tag{1}$$

where $N = \sum_{i=1}^{W}\sum_{j=1}^{H} M^{ij}$ is the number of black pixels ($M^{ij} = 1$) in the sketch image.

5 COLORIZATION

Although the existing method LBIE [Chen et al. 2018b] can be directly used to colorize a segmented object sketch or background region, it suffers from artifacts as shown in Section 8. There are two major limitations with LBIE. First, the architecture of its image encoder and decoder is not suitable for the sketch data or background regions. Second, its image-text fusion model is not effective, producing poor-quality results. For example, given an instruction *"the car is red with dark gray windows"*, LBIE failed to segment the car windows and colorized the car windows in red, as shown in Fig. 11. To achieve better colorization, we designed new architectures tailored for the characteristics of foreground and background regions for the tasks of both foreground and background colorization.

5.1 Foreground Colorization

Overview. As illustrated in Fig. 4, our network for foreground object instance colorization is essentially a generative adversarial network (GAN) consisting of a generator G and a discriminator D. Unlike traditional generators, such as pix2pix [Isola et al. 2017], only taking as input the single-modal visual image data, our generator G needs to model the interactions between the text description and visual information since the generated colorization results should be constrained by the text information.

Generator & Discriminator. The generator G consists of three modules: an image encoder which encodes the features of the $H \times W$ input sketch image of a segmented object instance generated from the instance matching stage, a fusion module which fuses the text information of a natural language expression into the image feature maps generated by the image encoder, and finally an image decoder which takes the fused features and produces an $H \times W \times 3$ output. We use the MRU blocks [Chen and Hays 2018] as the backbone of both the encoder and decoder modules. The MRU block, which was first proposed in [Chen and Hays 2018], takes an extra image input and produces new feature maps by dynamically merging the information in the extra image into the original input feature maps. It is proven that the MRU block has superior performance for sketch data over naive convolutional approaches. In our implementation, we use one convolution layer and four cascaded MRUs to encode the $H \times W$ input object sketch image into $H' \times W'$ feature maps ($H' = \frac{H}{32}, W' = \frac{W}{32}$) in the encoder. For the decoder, five MRUs are cascaded as a deconvolutional network. Skip-connections are applied between the encoder and the decoder. The fusion module is an RMI model similar to the one used in the instance matching stage (Fig. 3). It incorporates the text information into the $H' \times W'$ image feature maps, and outputs $H' \times W'$ fusion features. In our experiments, H and W were both set to 192.

The discriminator D takes as input the synthesized or ground-truth color image, and decides whether the image is fake or real. We also build the discriminator with the MRU blocks. D outputs two logits, one for GAN loss and the other for classification loss.

Loss Functions. Our GAN objective function is similar to the one in SketchyGAN [Chen and Hays 2018] and can be expressed as $L_{GAN}(D, G)$. As SketchyGAN, an auxiliary classification loss $L_{ac}(D)$ for D is introduced to maximize the log-likelihood by the predicted and ground-truth labels. The generator maximizes the log-likelihood as $L_{ac}(G) = L_{ac}(D)$ with the discriminator fixed but the image to be classified as a synthesized one.

Different from SketchyGAN, our task intends to generate colorization results with respect to user-specified instructions, instead of diverse results. We thus do not exploit the perceptual loss or the diversity loss. We replace the direct L1-distance in SketchyGAN with smooth L1-distance as the supervision loss, as a color term in our study might not correspond to a unique RGB value. For example, in Fig. 4, we might say the colors of both the synthesized image and the ground-truth are *"dark blue"* even though their corresponding RGB values are slightly different. Hence the penalty should be smaller than direct subtraction. We define the supervision loss as:

$$L_{sup}(G) = \begin{cases} \frac{1}{2}(G(x,s)-y)^2 & \text{if } \|G(x,s)-y\|_1 < 1 \\ \|G(x,s)-y\|_1 - \frac{1}{2} & \text{if } \|G(x,s)-y\|_1 \geq 1 \end{cases} \tag{2}$$

in which x is the input instance sketch image, y is the ground truth cartoon image, and s is the input text. $L_{sup}(G)$ is evaluated at each pixel and summed together to evaluate the loss for a whole image.

The complete loss functions of foreground instance colorization for discriminator D and generator G are defined as:

$$L(D) = L_{GAN}(D, G) + \lambda_1 L_{ac}(D), \tag{3}$$

$$L(G) = L_{GAN}(G) - \lambda_1 L_{ac}(G) + \lambda_2 L_{sup}(G). \tag{4}$$

where λ_1 and λ_2 are the coefficients. The discriminator D aims to maximize $L(D)$, while the generator G aims to minimize $L(G)$.

5.2 Background Colorization

Overview. As illustrated in Fig. 5, the proposed network for background colorization is a conditional GAN (cGAN) network, conditioned by a foreground image. It fills in the background regions of the input foreground image and produces a 768×768 high-resolution color image. The generator G consists of a colorization branch and a segmentation branch. The discriminator D is attached to the colorization branch and its design follows a general design of a discriminator in a cGAN [Isola et al. 2017] with minor adaptions. The whole network is trained in an end-to-end manner.

Generator & Discriminator. The colorization branch in the generator G has a similar structure to the network for foreground colorization. It shares the image encoder with the segmentation branch with an encoder-decoder structure. The image encoder in the generator G uses the residual block [He et al. 2016] units as its backbone. We choose the residual blocks here rather than the MRU blocks used for foreground colorization, as the residual blocks have superior capability in producing large-size smooth and gradual texture compared with the MRU blocks, and are thus more suitable for background

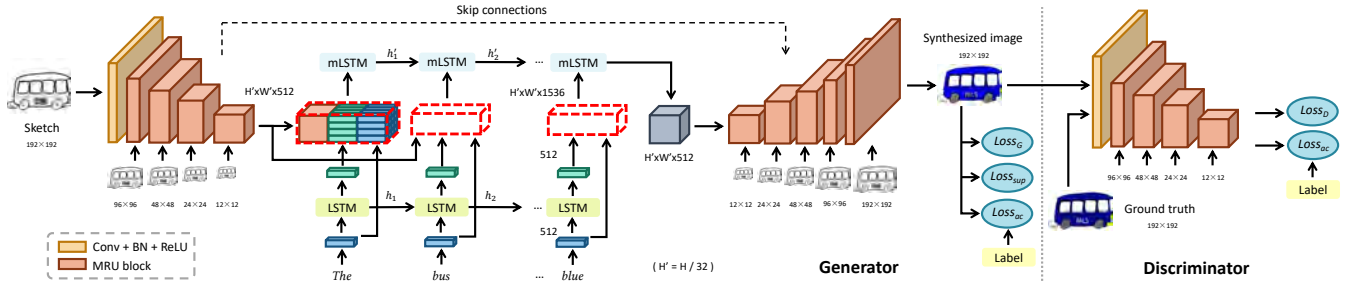

Fig. 4. Network architecture for foreground colorization. It is able to colorize objects from different categories. The generator has a U-Net architecture based on MRU blocks [Chen and Hays 2018], with skip connections between mirrored layers and an embedded RMI fusion module consisting of LSTM text encoders and multimodal LSTMs (mLSTM). This architecture is referred to as the *FG-MRU-RMI* network for conciseness below.

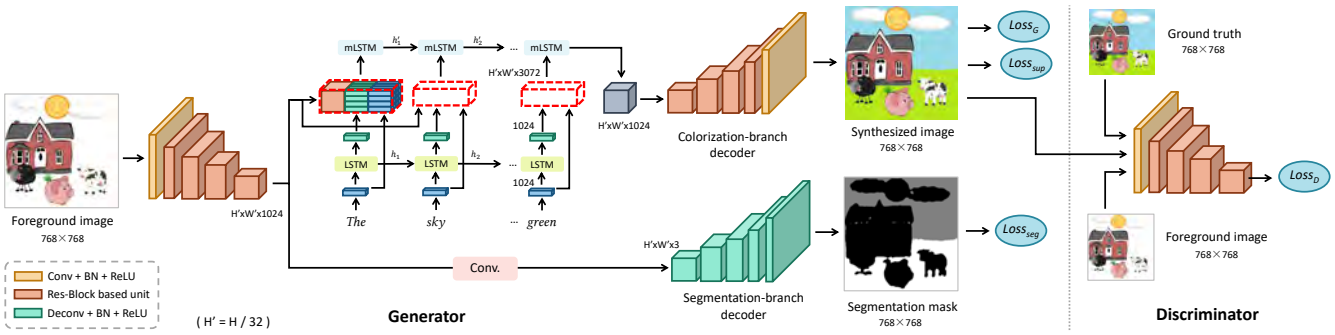

Fig. 5. Network architecture for background colorization, consisting of an image encoder built on residual blocks (Res-Block), a fusion module, a two-branch decoder, and a Res-Block based convolutional discriminator. This architecture is referred to as the *BG-RES-RMI-SEG* network.

colorization. The first unit of the image encoder employs a general convolutional layer, followed by four cascaded residual units, whose corresponding numbers of residual blocks are {3, 4, 6, and 3}, respectively. This structure is similar to ResNet-50. It is worth mentioning that the single colorization branch in fact performs implicit segmentation. Without the explicit segmentation branch, the generator G can still segment and colorize the background with the help of the mLSTM. The explicit segmentation branch is used to enhance segmentation results, producing more accurate segmentation boundaries, as shown in Fig. 15. The decoder of the segmentation branch, which produces the segmentation label map, is made up of a chain of general deconvolutional layers. The discriminator D takes the already colorized foreground image as the conditional input and decides whether a synthesized or ground truth color image is fake or not. It employs a Res-Block based image encoder like the one used in the generator G.

Loss Functions. The cGAN objective function has a similar form to the one in [Isola et al. 2017] and is expressed as $L_{cGAN}(D, G)$. There are three types of loss for the training of generator G: $L_{cGAN}(G)$, $L_{seg}(G)$, and $L_{L1-sup}(G)$. $L_{cGAN}(G)$ is the conditional GAN loss, and $L_{seg}(G)$ is the segmentation loss with the form of cross entropy, while $L_{L1-sup}(G)$ provides the supervision to the network with the L1-distance between generated images and ground truth images.

The background colorization model fills the background regions with color while leaving the foreground objects unchanged. Therefore, $L_{L1-sup}(G)$ does not need to take into account the penality of foreground object regions. We therefore propose a training strategy, referred to as *w/o-FG*, which ignores the contribution of the foreground objects in computing the supervision loss. We use a binary mask M^{ij} to divide the foreground and background regions ($M^{ij} = 1$ indicates a background pixel; otherwise, a foreground pixel). Given the paired image data, i.e., the input foreground image x and the ground truth image y, and the natural language description s, the supervision loss is then formulated as follows:

$$L_{L1-sup}(G) = \frac{1}{N}\sum_{i=1}^{W}\sum_{j=1}^{H}\left(M^{ij} * \|G(x,s)^{ij} - y^{ij}\|_1\right), \tag{5}$$

where $N = \sum_{i=1}^{W}\sum_{j=1}^{H} M^{ij}$ is the number of background pixels in the input image x.

In summary, the loss functions of background colorization for discriminator D and generator G are formulated as

$$L(D) = L_{cGAN}(D, G), \tag{6}$$

$$L(G) = L_{cGAN}(G) + \lambda_1 L_{seg}(G) + \lambda_2 L_{L1-sup}(G), \tag{7}$$

where λ_1 and λ_2 are the coefficients. See Section 1.1 in the supplementary material for more details about the loss functions.

Fig. 6. Illustration of the dataset for instance matching. Each foreground object instance has a ground-truth binary mask (in colors) and a corresponding natural language expression with optional location information. Random colorization information (in blue, e.g., *"light green"*) is included in the expression, following a set of sentence patterns, to support both instance matching and foreground colorization using the same expressions in the testing stage.

6 DATASETS

We built the datasets for training and testing our networks mainly on the SketchyScene dataset [Zou et al. 2018], which provides the instance and semantic segmentation ground truth for more than 7k scene sketch templates. Below we briefly describe each dataset (for more details, please see Section 2 in the supplementary material).

6.1 Data for Instance Matching

As SketchyScene has provided the instance segmentation ground truth for a large number of scene sketches, theoretically we only need to prepare the associated language expressions including the optional location information of each object instance, e.g., *"the tree in the middle"*, in the sketches to train the instance matching network. However, in practice, a user might specify an object of interest with location information and colorization goal within a single expression. To address this issue, we need to provide complete text-based instructions including the object category, optional location information, and colorization goal, e.g., *"the tree in the middle is dark green"* for each object instance.

Manually preparing user instructions for each object instance in a large-scale set of scene sketches requires a huge amount of user labour. We therefore turned to designing a fully automatic rule-based algorithm, which tries to imitate human cognition and expressing habits, to generate the user instructions. Specifically, the algorithm first automatically generated a phrase about an object and its location information (if necessary), e.g., *"the tree in the middle"*/ *"all the clouds"*, for each object instance. Afterwards, following a set of sentence patterns like "...is/are...", "...has/have...", the algorithm randomly selected a colorization description (e.g., *"dark green"*) from the instance colorization dataset (see below) and then attached it to each phrase to generate a complete instruction. Although we only designed a limited number of sentence patterns for each category of foreground objects, as shown in Section 7, the model trained on this dataset could still support the expressions within a large degree of flexibility.

We collected 38,557 pairs of object instance segmentation mask and language expression in total through the above automatic algorithm. These data were split into three sets: 30,094 pairs for training, 2,372 pairs for validation, and 6,091 pairs for testing, which covered $1,337 + 116 + 242 = 1,695$ scene sketches from the SketchyScene dataset. The object instances were from 24 categories as shown in Table 1. This dataset is referred to as *MATCHING* dataset below. Several representative samples are illustrated in Fig. 6.

Fig. 7. Illustration of the dataset for foreground instance colorization. 4,587 triplets of cartoon image, edge map (sketch), and text description were collected. The location information (in blue) was randomly added to color descriptions to imitate the location-related instructions users might assign to the system.

6.2 Data for Foreground Instance Colorization

The data for foreground instance colorization includes color objects, the corresponding edge maps (sketches), and the language descriptions regarding the color information of those color objects. To collect these data, we crawled cartoon instance images from the Internet and then leveraged X-DoG [Winnemöller 2011] to extract an edge map as the corresponding sketch for each cartoon image. We employed 6 subjects to manually create color annotations for each cartoon image and then produced a color description automatically for each cartoon image based on pre-defined sentence structure patterns. In this way, we built a collection of in total 4,587 triplets of cartoon image, edgemap, and language description, covering the same 24 object categories as the data for instance matching. These data were split into two sets: 3,822 triplets for training, 765 triplets for validation. Moreover, to evaluate the performance of our instance colorization model on real sketches rather than edgemaps, we also built a test set which includes 1,734 pairs of instance sketches and descriptions. These instance sketches which cover all the 24 object categories were extracted from the scene sketches of the test set in SketchyScene. All the cartoon images and edgemaps/sketches were resized to 192×192. The descriptions cover 16 different colors. The number of colors for each object instance varies from 1 to 3. The number of words in each description varies from 4 to 15, with an average of 6. We randomly added the location information based on pre-defined sentence structure patterns for each collected description. This dataset is referred to as *FOREGROUND* dataset below. Several representative samples are shown in Fig. 7.

6.3 Data for Background Colorization

Four modality data, namely, foreground image (with empty background), color image (with color background), text description, and segmentation label map are required for background colorization. To collect such data, we composed images of foreground objects by placing color object instances into 768×768 white images following the object layout information provided by the sketch templates from the SketchyScene dataset. We then employed 24 users to produce the color images (by manually painting the background regions

Table 1. Details for each category of the dataset for instance matching.

	bench	bird	bus	butterfly	car	cat	chick	cloud	cow	dog	duck	grass	horse	house	moon	person	pig	rabbit	road	sheep	star	sun	tree	truck	Total
Train	605	1725	716	343	1205	685	836	1886	580	1254	958	1941	184	2209	56	2655	188	647	744	810	62	1068	8504	233	30094
Val	15	77	32	10	71	64	54	160	102	93	76	195	22	88	6	139	12	97	50	140	14	64	778	13	2372
Test	94	188	36	48	146	86	178	368	105	218	156	347	80	776	50	845	66	72	118	159	57	182	1651	65	6091

with solid colors). For simplicity, we considered only two types of background: sky and ground. The segmentation information and the corresponding text-based color descriptions for the background regions were obtained by examining different painting colors within the background regions. For each foreground image, we augmented the data by altering the colors of background regions in the user painting to generate three additional pairs of color image and description. In Fig. 8, we illustrate a representative sketch template, the corresponding foreground image, the color image with user-painted background, segmentation map, and 3 pairs of augmented color image and text description. With such data augmentation, we in total obtained 15,728, 1,200, and 2,908 quadruple data for training, validation, and testing, respectively. The text descriptions totally cover 11 colors. The number of words in each description is 9 on average. This dataset is referred to as *BACKGROUND* dataset below.

7 INSTANCE MATCHING EXPERIMENTS

7.1 Ablation Studies

We validated the design choices of our proposed instance matching model in three main aspects: feature extraction, text-image fusion, and the effect of training with/without background (for more implementation details, see Section 1.2 in the supplementary material).

Feature Extraction. For sketch feature extraction, we evaluated three alternatives: **FCN-8s** [Long et al. 2015], **SegNet** [Badrinarayanan et al. 2017] and **DeepLab-v3+** [Chen et al. 2018c]. These alternatives are the most effective networks for image feature extraction. FCN-8s is empowered by combining both coarse and fine image features from different layers during upsampling to allow the network to discriminate more low-level features (e.g., shape and boundary). SegNet improves its feature extraction capability by leveraging an indices-based unpooling scheme for decoding to form a symmetrical encoder-decoder architecture, which improves the upsampling performance. To enlarge the fields of view of filters, DeepLab-v2 employed by DeepLabv2-RMI learns image features by exploiting atrous convolution and atrous spatial pyramid pooling (ASPP). Inherited from Deeplab-v2, Deeplab-v3+ further enhances image feature learning ability by augmenting the ASPP module with image-level features which contain contextual information and by replacing the bilinear upsampling process with a decoder.

Text-image Fusion. We studied how different cross-domain fusion mechanisms perform on this task. We validated the **RMI** model employed in DeepLabv2-RMI with two alternatives: an attention-aware RMI model (**RMI-Attn**) and a recurrent attention injected RMI model (**RMI-RAttn**). In the RMI-Attn mechanism, we presumed that hidden states from the intermediate mLSTMs, rather than merely the last one as in RMI model, might contribute more to the final multimodal feature. This presumption was based on our observation that some words at the end of a sentence, e.g., *"The person on the right has brown hair and is in purple shirt"*, are not relevant to the segmentation. We then introduced a shared fully connected layer f to learn a word-level attention and used it to re-weight the hidden states from all time steps to obtain an attention-aware multimodal feature (i.e., RMI-Attn produces the attention only when the last time step finishes).

RMI-RAttn is another alternative fusion scheme, where word-level attention is used in a recurrent manner as LBIE [Chen et al. 2018b]. The major difference between RMI-RAttn and RMI-Attn is that the attention is computed based on the hybrid of word features and input multimodal features (from each time step) within each image region at every time step recurrently. At each time step of RMI-RAttn, the attention produced within a certain image region indicates the importance of each word to that region. Then, an attentive feature map, generated by incorporating the attention and the word features, is fed to mLSTM as input. The multimodal feature map used to generate the segmentation mask of the target object instance(s) is achieved at the last time step.

Training with/without Background. We also validated the effect of the *ignoring background* training strategy we formulate in Eq. 1. Apart from this training strategy as presented in Section 4, we trained the DeepLabv2-RMI model by considering the contribution of all the $W \times H$ pixels. We refer to the former as **w/oBG** and the latter as **withBG**. We next present our evaluation of all the above models on both the validation and test sets in the *MATCHING* dataset.

7.2 Results

Quantitative Results. Following Mask R-CNN [He et al. 2017] and RMI [Liu et al. 2017a], we use mask IoU and mask AP, AP_{50} and AP_{75} as metrics to measure the segmentation accuracy of each comparison model. We summarize the comparison results in Table 2, where we can see the DeepLabv2-RMI architecture with the *ignoring background* training strategy achieves the best overall performance.

From the study of feature extraction we can see that FCN-8s and SegNet perform worse than DeepLab-v2 on most of the metrics. This indicates that their upsampling schemes, i.e., combining coarse and fine features or using indices-based unpooling, do not suit sketch images well. DeepLab-v3+ performs slightly worse than DeepLab-v2 on most of the metrics, possibly because it fails to extract enough contextual information into the image-level features due to the sparsity of sketch images.

From the study of text-image fusion, we can see that the RMI-Attn model is close to but slightly weaker than the RMI model. This may be because the former is forced to emphasize particular

Fig. 8. Illustration of the data collection for background colorization. The three right-most columns are three pairs of color image and text description generated by automatically altering the colors for background regions in the user painting.

words while paying less attention to other words and thus might lose important information for segmentation. In our task, a target object could be constrained by other reference information like relative position and reference object (e.g., "the car in front of the house"). Assigning too much weight to particular words (like "car" and "house") might result in inaccurate segmentation due to the lack of contextual information. As for the RMI-RAttn model, its accuracy drops by 8 – 12% over the RMI model. This does not agree with the experimental results in referring image segmentation works such as [Shi et al. 2018]. This is possibly because, unlike natural images, the contextual information within each image region in the feature map of sketch images is too weak to provide strong constraints for the attention learning, thus making it difficult to match the correct text information with image regions.

By comparing the **withBG** model with **w/oBG**, we can see that the performance of **withBG** drops sharply on all the metrics, indicating that the *ignoring background* training strategy indeed has significant effects on referring image segmentation, as it does on the task of semantic segmentation in SketchyScene [Zou et al. 2018].

Qualitative Results. Our visual comparisons confirm the results indicated in Table 2. In Fig. 9, we show representative results of the comparison methods on a typical scene sketch from the test set. We can see that our DeepLabv2-RMI model generally produces the results close to the ground truth.

By comparing the results of all the feature extractors, we can see that both FCN-8s and SegNet perform worse than DeepLab-v2 when the input instructions are complex. For example, given the instruction *"the bench in front of the house is brown"* in Fig. 9 (Bottom) where another word *"houses"* is used as the reference of the target object *"bench"*, both FCN-8s and SegNet fail to segment the target object. The visual quality of DeepLab-v3+ is close to that of DeepLab-v2.

By observing the results from the text-image fusion study, we can see neither RMI-Attn nor RMI-RAttn can successfully segment the target objects in all cases. Specifically, RMI-Attn fails to understand the complex instructions (e.g., Fig. 9 (Bottom)) due to the unreliably learned attention. By contrast, RMI-RAttn can understand the complex instruction but fails to segment multiple objects (e.g. the "grasses" in Fig. 9 (Middle Row)). It is possibly because the incorrect attention makes the model hard to associate the text information with all the corresponding regions in the image correctly. From Fig. 9, it can be observed that the **withBG** model performs poorly and can hardly segment any target object due to the poor capacity of sketch image feature extraction.

Table 2. Quantitative comparison of text-based instance segmentation performances (mask IoU & mask AP) from different alternative approaches on our *MATCHING* dataset.

Model	mIoU		mask AP					
	val	test	val			test		
			AP	AP_{50}	AP_{75}	AP	AP_{50}	AP_{75}
Feature extraction								
FCN-8s	70.83	**78.05**	41.77	64.06	49.17	44.28	68.56	51.70
SegNet	78.62	71.83	46.29	71.68	54.11	44.21	67.66	51.98
DeepLab-v3+	83.27	76.07	46.44	72.47	53.77	45.07	69.36	52.86
DeepLab-v2	**83.67**	75.90	**47.04**	**73.09**	**54.72**	**45.97**	**70.79**	**53.90**
Text-image fusion								
RMI-Attn	83.52	75.52	46.94	73.05	54.59	45.35	69.94	53.15
RMI-RAttn	73.53	66.25	38.60	60.36	44.55	37.82	57.97	44.60
RMI	**83.67**	**75.90**	**47.04**	**73.09**	**54.72**	**45.97**	**70.79**	**53.90**
Training with/without background								
withBG	26.82	28.75	5.84	7.62	7.16	5.96	7.94	7.19
w/oBG	**83.67**	**75.90**	**47.04**	**73.09**	**54.72**	**45.97**	**70.79**	**53.90**

7.3 Generalization Study

We evaluated the generalization ability of the instance matching model. From the collected data in the user study of our system's overall performance (Section 8.3), we found that some language instructions were clearly beyond the coverage of the sentence patterns of the training data. Apart from expected short-phrase instructions like those in Fig. 1, there are mainly four other cases as shown in Fig. 10: 1) *Part-Aug*: instructions with descriptions about object parts (e.g., "tires" and "wheels") which never exist in the training data; 2) *Gram-Err*: instructions with grammar errors; 3) *Multi-Cate*: instructions involving objects from multiple categories; and 4) *Alt-Name*: instructions with alternative category names (e.g., "taxi") which do not exist in the training data.

We observe that, besides expected short phrases, the matching model can still successfully segment target objects in the cases of

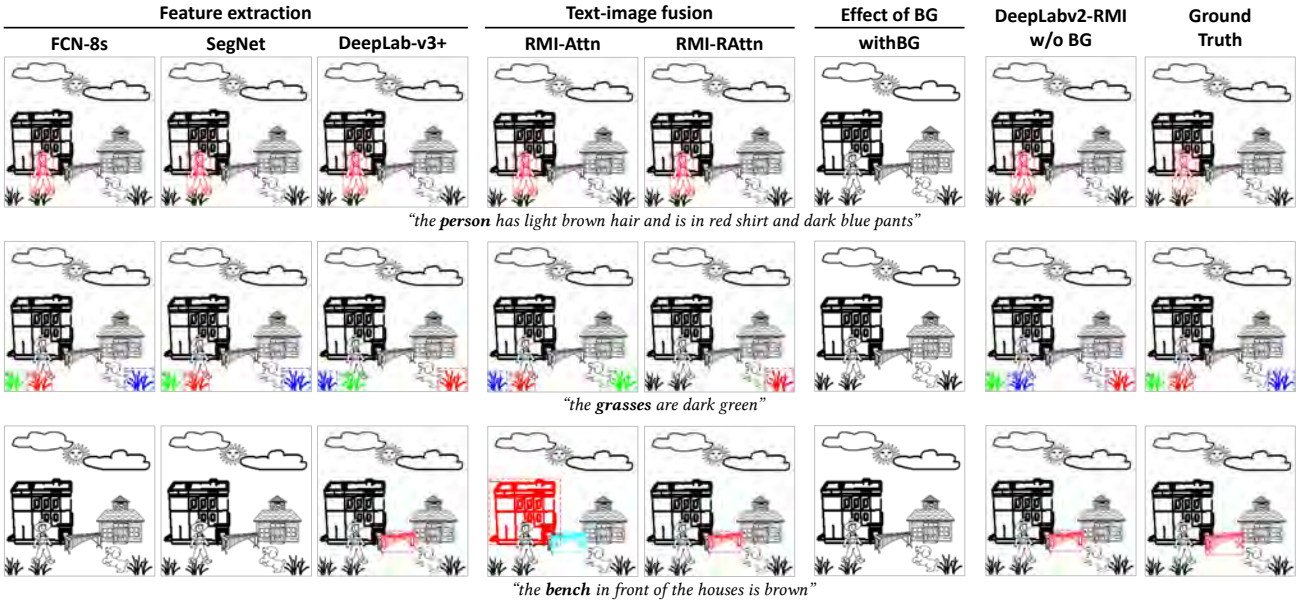

Fig. 9. Representative results of alternative approaches on a typical sketch in the *MATCHING* dataset.

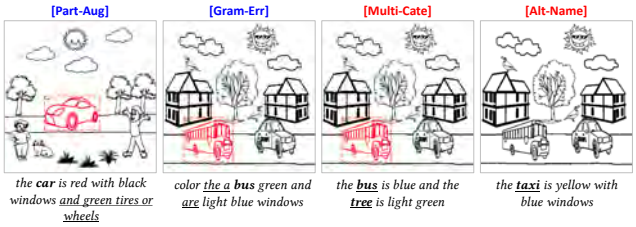

Fig. 10. Generalization study for instance matching on instructions in the wild.

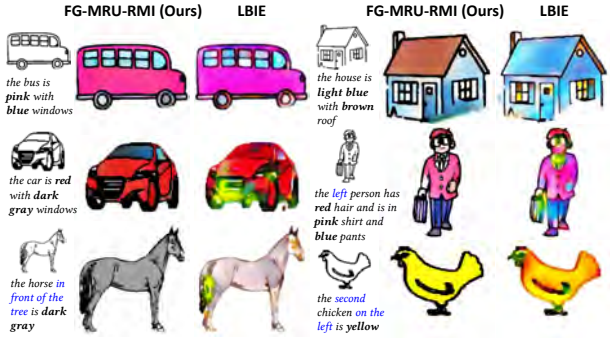

Fig. 11. Network comparison for foreground object instance colorization: FG-MRU-RMI *vs.* LBIE [Chen et al. 2018b].

Part-Aug and *Gram-Err*. As for the case of *Multi-Cate*, the matching model only segments target object(s) described by the words in the first part of an instruction in most testing cases. (e.g., only the "bus" is segmented in the example of *Multi-Cate* in Fig. 10, and only the "tree" is segmented when given the expression *"the tree is light green and the bus is blue"* where "tree" and "bus" were presented in reverse order). This can be explained by the fact that our training dataset does not include this kind of instructions with a different sentence structure. This causes confusion for the model when it sees such *wild* instructions. In the case of *Alt-Name*, the model can hardly segment any object because the alternative object name is not understood at all and is regarded as an *unknown* word. With the lack of the most important information, it is difficult for the model to distinguish the target according to the rest of the information in the sentence.

8 COLORIZATION EXPERIMENTS

In this section, we first study the alternatives for the foreground object instance colorization network, i.e., FG-MRU-RMI, and the background colorization network, i.e., BG-RES-RMI-SEG. Afterwards, we describe a user study we conducted to investigate the faithfulness and effectiveness of the proposed colorization networks. We finally study the overall performance of the proposed system and investigate its generalization and usability (for more implementation details, please see Section 1.2 in the supplementary material).

8.1 Alternative Studies

8.1.1 Foreground Instance Colorization. We first study the effectiveness of our foreground instance colorization solution, FG-MRU-RMI, in comparison with LBIE [Chen et al. 2018b], which is the

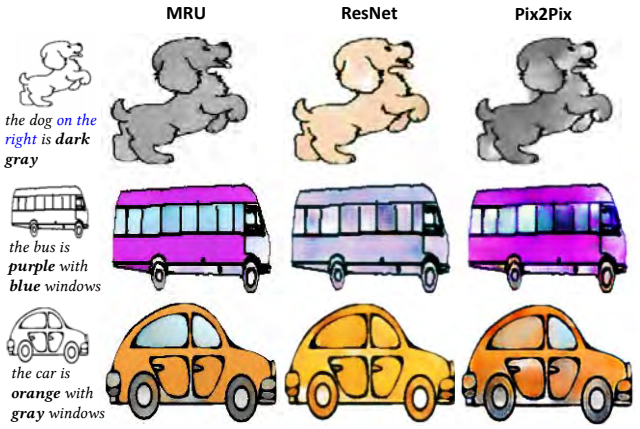

Fig. 12. Building block (backbone) comparison for foreground instance colorization: MRU (the adopted one) *vs.* ResNet *vs.* Pix2Pix.

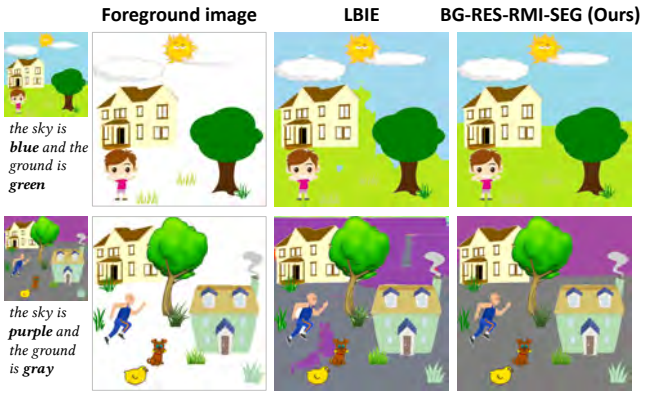

Fig. 13. Network comparison for background colorization: LBIE *vs.* BG-RES-RMI-SEG.

state-of-the-art language-based automatic image colorization network. Then we investigate what kind of backbone (building block) is the most effective. We mainly compare the employed MRU blocks with ResNet blocks [He et al. 2016] and Pix2Pix blocks [Isola et al. 2017], which are the building blocks widely used in deep models for various applications. We trained all the comparison models on the same training set of the *FOREGROUND* dataset. For fair comparison, all the hyper-parameters for these models were set to be equivalent.

FG-MRU-RMI vs. LBIE. We tested the two methods on all the examples from the test set of the *FOREGROUND* dataset. In Fig. 11, we show several representative results. It can be seen that FG-MRU-RMI significantly outperforms LBIE in the aspect of *faithfulness*, i.e., whether the colorization results are consistent with the language descriptions. For instance, given a user instruction *"the car is red with dark gray windows"* and a car sketch, in the LBIE result there is an undesired green region in the front part of the car, and the car windows are not in dark gray. By contrast, the FG-MRU-RMI result meets the user's colorization requirement. In the aspect of *effectiveness*, which measures the overall visual quality of the colorization results, FG-MRU-RMI is also superior to LBIE. For example, color bleeding artifacts are more apparent in the LBIE results.

From the results, e.g., the yellow chicken in the bottom row and the light blue house in the top row, we can also see that both FG-MRU-RMI and LBIE are capable of understanding the user-specified color information reasonably well (i.e., associating a specific color with the color described in the text description). However, when an instruction contains different colorization requirements for different object parts, e.g., *"the house is light blue with brown roof"*, LBIE often fails to assign the right colors to the corresponding object parts (e.g., the roof of the house is colorized with blue by LBIE). By contrast, FG-MRU-RMI achieves better results in such scenarios, indicating that FG-MRU-RMI is more capable of inferring the part-level segmentation of an object-level sketch.

MRU vs. ResNet vs. Pix2Pix. With the same architecture shown in Fig. 4, we built two alternative networks for FG-MRU-RMI by replacing the MRU blocks with ResNet or Pix2Pix blocks, respectively. We then compared these three networks using the test set of the *FOREGROUND* dataset. In Fig. 12, we show several representative results where we can see that MRU achieves relatively better performance than ResNet in the aspect of faithfulness (e.g., see the dog example). The results generated by MRU also have better visual quality than those by Pix2Pix, e.g., the color of the bus windows has less bleeding in the results of MRU.

8.1.2 Background Colorization. We first compare our BG-RES-RMI-SEG network in Fig. 5 with LBIE in the task of background colorization. Then we compare the performance of the three building blocks, i.e., MRU, ResNet, and Pix2Pix. Lastly, we present our ablation study on the effects of the segmentation branch of BG-RES-RMI-SEG and the w/o-FG training strategy (i.e., excluding the contribution of foreground) for the computation of the supervision loss. We trained all the compared models on the training set in the *BACKGROUND* dataset. All the hyper-parameters for the comparison models were set to be identical to result in fair comparisons.

BG-RES-RMI-SEG vs. LBIE. We compared LBIE and BG-RES-RMI-SEG on all the examples from the test set of the *BACKGROUND* dataset. In Fig. 13 we show the results from two representative examples, from which we can see that both LBIE and BG-RES-RMI-SEG can colorize most background regions with correct colors. This indicates that both LBIE and BG-RES-RMI-SEG have the ability to understand instructions, do segmentation, and paint with the required colors. Relatively, BG-RES-RMI-SEG is superior to LBIE. This can be observed from the green area on the right side of the house (the upper example) and the purple regions close to the dog (the bottom example) in the LBIE results. The better performance of BG-RES-RMI-SEG is mainly because of the segmentation branch and the stronger multimodal feature learning strength brought by the RMI module and ResNet block chain.

MRU vs. ResNet vs. Pix2Pix. Similar to foreground colorization, we built two alternative networks for BG-RES-RMI-SEG by replacing its ResNet blocks with MRU or Pix2Pix blocks, respectively. For fair comparison, the w/o-FG strategy was used for all the three networks.

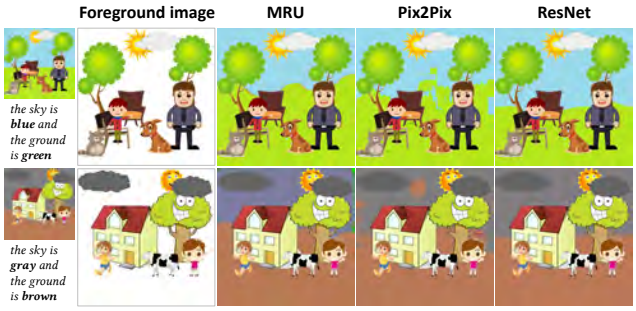

Fig. 14. Backbone comparison for background colorization: MRU *vs.* Pix2Pix *vs.* ResNet (the adopted one).

In Fig. 14 we show the results of the comparison networks on two representative examples. ResNet outperforms the alternatives. It can be easily seen that Pix2Pix produces obvious artifacts, possibly because the cascaded 5-layer Pix2Pix building blocks are too shallow for high-resolution images (768×768 in our experiments). MRU performs better but still suffers from artifacts around the boundaries of different semantic regions. This might be because the MRU blocks are particularly proposed for inferring color images from sketches rather than painting large blank regions in our scenario.

Ablation Study. To investigate the effects of the explicit segmentation branch of the architecture shown in Fig. 5 and the w/o-FG training strategy, apart from **BG-RES-RMI-SEG**, we built three additional comparison models, including **w/o SEG**: the model without the segmentation branch; **with FG**: the model involving foreground objects in the supervision; **w/o SEG - with FG**: the model without the segmentation branch but with the contribution of foreground objects considered. We compared these four models on all the examples from the test set of the *BACKGROUND* dataset.

The comparison results of a representative example is shown in Fig. 15. By pair-wisely comparing the results generated by **w/o SEG** and ours, and those generated by **with FG** and **w/o SEG - with FG** we can see that, without the segmentation branch, the models produce less reasonable boundaries between the regions with different semantic labels (i.e., sky and ground). We can also see that, without the w/o-FG training strategy, the same architecture produces artifacts around the boundaries of the foreground objects (see the boundaries of the clouds in the results of **with FG** and **w/o SEG - with FG**). By contrast, our proposed model achieves the best performance. This conclusion will be further verified by our user studies.

8.2 User Studies

Study Settings. To further quantitatively evaluate the colorization results of the alternative experiments in Section 8.1, we designed and conducted two on-line user studies, *effectiveness study* and *faithfulness study*. The effectiveness score reflects the capability of the image encoder and decoder in the compared networks. The faithfulness score can be affected by the capabilities of both the text-image fusion module and the image encoder-decoder structure.

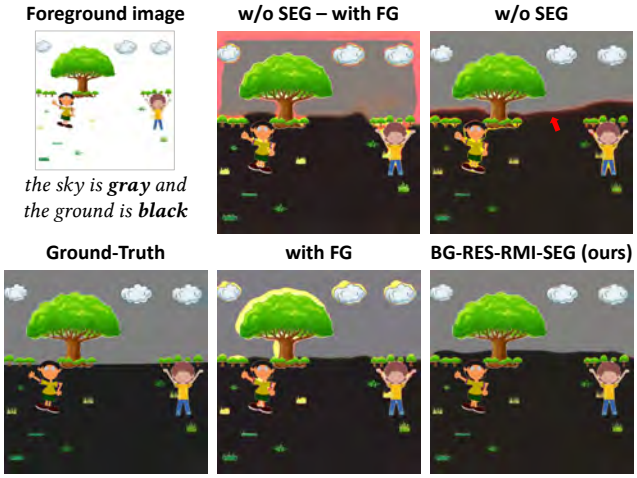

Fig. 15. Illustration of the advantages of the segmentation branch of the proposed architecture and the w/o-FG training strategy for background colorization.

We recruited 26 participants for both of the studies. All participants had no prior knowledge of this project. In both studies, for each test sample, each participant was shown a page of the input data and the corresponding colorization results of comparison approaches in random order. In the faithfulness study, the participants were required to pick out the color image which best meets the colorization requirement in the given language expression. In the effectiveness study, the participants were required to pick out the most visually pleasing color image. For foreground colorization task, questionnaires were generated on 72 sets of randomly selected test samples covering 24 categories. For background colorization task, 40 randomly selected sets of test samples were used for the study.

For foreground colorization task, 18 valid study results were completed for both sets of the experiments (i.e., FG-MRU-RMI vs. LBIE, and MRU vs. ResNet vs. Pix2Pix) in both the faithfulness and effectiveness studies. $(18 + 18) \times 72 = 2,592$ trials were collected in total for either of the faithfulness and effectiveness studies. For background colorization task, 26, 22, and 20 valid study results were returned for the three sets of experiments, respectively: BG-RES-RMI-SEG vs. LBIE, MRU vs. ResNet vs. Pix2Pix, and the ablation study. In total, $(26 + 22 + 20) \times 40 = 2,720$ trials were collected for either of the faithfulness and effectiveness studies.

Results. The statistic results of the collected study data are presented in Fig. 16 and Fig. 17. We can see that the proposed FG-MRU-RMI clearly outperforms LBIE in foreground colorization task when measuring in both faithfulness and effectiveness. Relatively, BG-RES-RMI-SEG has more comparative advantages over LBIE in background colorization task. This is mainly because both the image encoder and decoder (49 layers) in BG-RES-RMI-SEG are much deeper than those in by LBIE (15 layers), which enables BG-RES-RMI-SEG to be more capable of dealing with relatively large-size images (768×768 for background colorization in our implementation). A similar reason can also be used to explain the backbone comparison

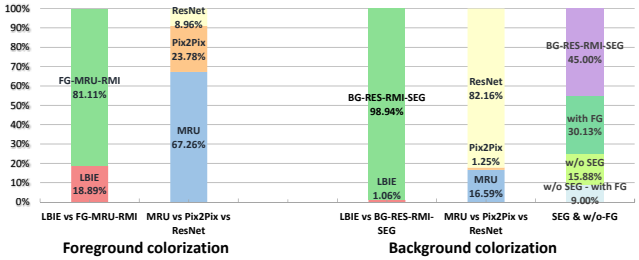

Fig. 16. Statistic results for faithfulness study.

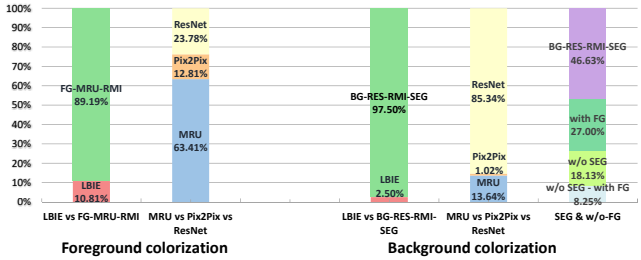

Fig. 17. Statistic results for effectiveness study.

results in the background colorization task, where ResNet (49-layer) beats MRU (13 layers), which in turn beats Pix2Pix (5 layers).

By comparing the relative advantages of each backbone alternative on the foreground and background colorization tasks, we conclude that MRU is more suitable for foreground colorization (inferring color images from sketches) while ResNet is more suitable for background colorization (inferring color images from blank regions). By comparing the statistic results of **w/o SEG** and **with FG**, i.e., 15.88% vs. 30.13% (faithfulness) and 18.13% vs. 27.00% (effectiveness), we can see that the system benefits more from the explicit segmentation branch than from the w/o-FG strategy.

8.3 Overall Performance

User Study Settings. We conducted two additional studies to evaluate the overall performance of the proposed system: *targeted colorization* where participants were required to colorize a sketch into target color images as closely as possible, and *un-targeted colorization* where participants were allowed to colorize a sketch with free instructions. Before the experiment we selected 20 scene sketches with reasonable instance segmentation results from the test set of SketchyScene [Zou et al. 2018] and generated a target color image for each sketch in the same way as that we prepared for the *BACKGROUND* dataset. In this way, we obtained 20 pairs of sketch and target color image for the study of targeted colorization, and 20 sketches for the study of un-targeted colorization. Afterwards, we invited six participants to provide the input expressions for the un-targeted colorization study first and then for the targeted colorization study. These participants included a 10-year-old boy in primary school, a 14-year-old boy in high school, two female and one male graduate students aged 21 to 23, and a 30-year-old female working in a company. The 10-year-old boy and one of the female graduate students are native English speakers, while the others are not. Each of the 20 examples was randomly assigned to 2 participants to conduct the studies.

In the un-targeted colorization study, before the description data collection we only showed the participants Fig. 1 for illustrating the workflow of our system together with the color types of the foreground and background objects in the training data as the tutorial. By contrast, in the targeted colorization study, before the data collection, we additionally showed them the category names of all the foreground and background objects, together with Fig. 7 and Fig. 8 for showing the examples of sentence patterns in the training data. The participants were required to provide complete instructions to colorize the entire sketch for each example in both studies.

User Study Results. For each study, we collected instructions for $2 \times 20 = 40$ sketches in total, each with around 10 instructions. With these input data, we generated $40 \times 10 = 400$ sets of progressive results. For each instruction, our system returned a colorization result within 2 seconds. Below we briefly present some of the results (for more results, please see Section 3 in the supplementary material).

In the un-targeted colorization study, since the participants were only shown a limited number of sentence patterns in Fig. 1 and were not informed of the object category names in the training data, we collected a large proportion of instructions which were beyond the coverage of the training data (see Fig. 18) in terms of *"wild" sentence structure* (e.g., the most similar training instruction to "yellow road" (A4) is "the road is yellow"), *language grammar* (e.g., "the clouds are are in dark gray" (B4)), and *unsupported words* (e.g., the verb "colorize" in "colorize the bus in purple" (B5) does not exist in the training data). However, our system still produces reasonable results with these kinds of input data. Fig. 18 (Bottom) also shows there is no need to colorize the foreground before the background (B3), and our system supports re-colorization (B5).

In the targeted colorization study, even though we showed the participants some sentence patterns in our training dataset, we still found diverse expressions in their instructions. In Fig. 19, we show the results of two representative sets of input data from four users. These instructions include some significantly different expressions or slightly different color goals towards the same target. Take the left case for an example: user A described the objects briefly while user B preferred to describe in more detail, e.g., "with orange flame" for the "sun". Moreover, for the "house", the sentence structures between user A and B were quite different. Despite the expression diversity, our system still produces color images close to the target cartoon for the two users. In the right example, we can see that different users assigned different but visually close colors probably due to their individual color cognition, e.g., "light brown hair"/"purple pants" from user C and "dark brown hair"/"black pants" from user D. Our system produces results that are slightly different but still consistent with the target images to some extent.

Generalization. To further evaluate the generalization ability and robustness of our system, we also applied our system to a number of wild sketch images from the Internet. These test sketch images included three types of styles: cartoon-style drawing, artist free-hand drawing, and anime line art. In addition, we also evaluated our

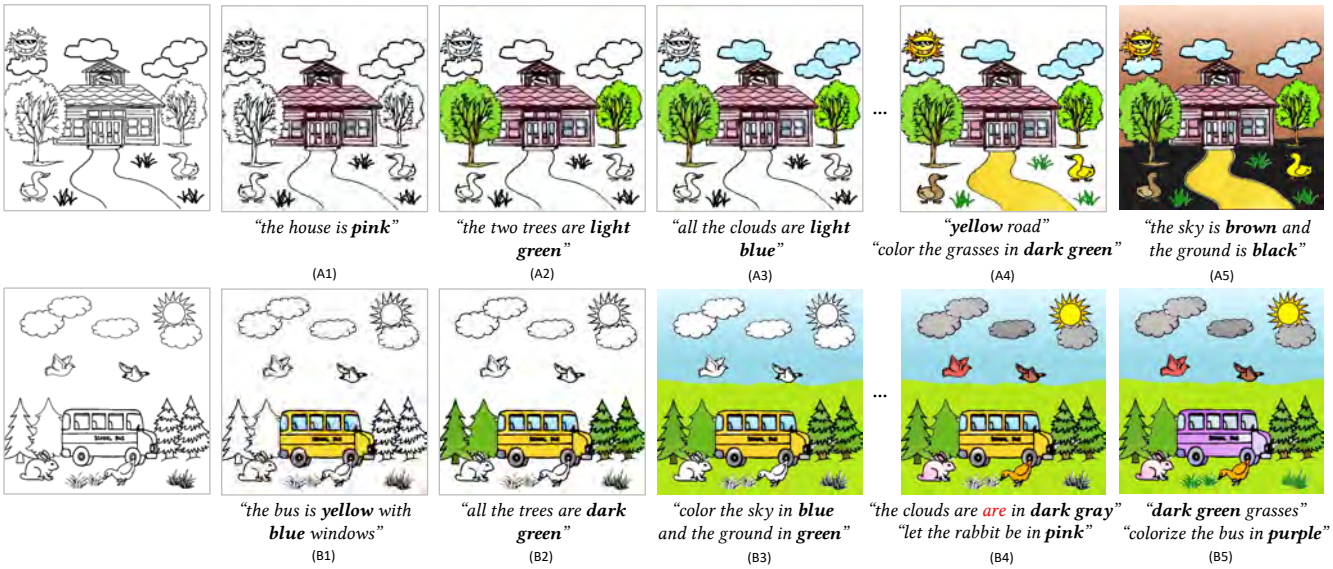

Fig. 18. Two representative interactive colorization results in the un-targeted colorization study.

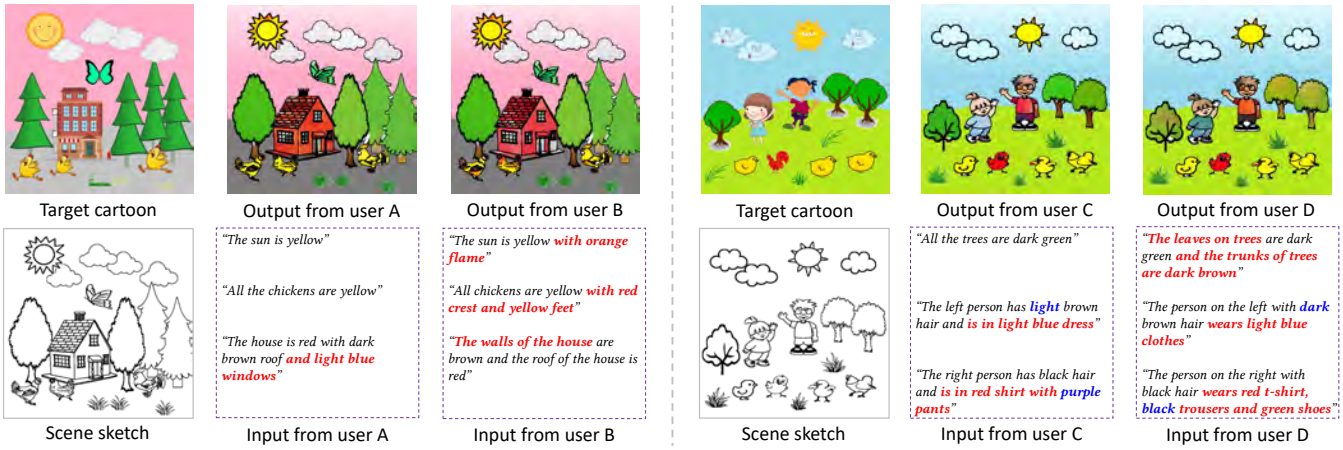

Fig. 19. Representative results from the targeted colorization study. For better visualization, we highlight the different *expressions* towards the same target in red and different *color goals* in blue.

system on some non-artist freehand sketches in the Sketchy [Sangkloy et al. 2016] (instance-level) and Photo-Sketching [Li et al. 2019] (scene-level) datasets, which are more *"sketchy"* than the three styles just mentioned. We manually constructed the colorization instructions for these wild sketches. Fig. 20 and Fig. 21 illustrate the results, from which we can see that our system is also capable of generating desirable results.

9 CONCLUSION AND DISCUSSIONS

Empowering machines with the ability to intelligently understand both natural language expressions and scene sketches with various styles and to perform colorization tasks is a challenging but useful task. This paper for the first time presents a language-based system for interactive colorization of scene sketches. This is enabled by multiple carefully designed deep networks for instance matching, foreground colorization, and background colorization, as well as dedicated large-scale datasets for network training. Our comprehensive experiments have demonstrated the effectiveness and robustness of the presented system. We are interested in combining a language-based interface and other interfaces (e.g., scribble-based) for a more powerful multimodal colorization system. Our current system can still be improved in multiple aspects, as detailed below:

Lack of Language Generality for Instance Matching. One major limitation of our system is the lack of generality of the language component and the compatibility with human's linguistic habits. In

Fig. 20. Generalization study on the wild sketch data , including: (a) & (b) cartoon-style drawings, (c) artist freehand drawing and (d) anime line art.

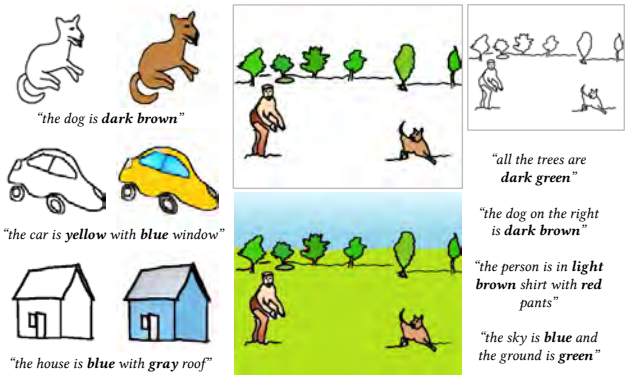

Fig. 21. Generalization study on non-artist freehand sketches. The instance-level sketches are from the Sketchy [Sangkloy et al. 2016] dataset and the scene-level one is from the Photo-Sketching [Li et al. 2019] dataset.

our targeted colorization study (Section 8.3), most participants said that based on their linguistic habits, they preferred to describe multiple objects even of different categories every time, rather than one by one as required in the study, like *"The sun is orange and the clouds are light blue"*. As shown in Fig. 10 and the analysis in Section 7.3, it is still difficult for our system to fully understand this kind of sophisticated expressions with multiple categories. Moreover, some participants also pointed out that, for example, in the right case of Fig. 19, after describing *"The left person has ..."*, they tended to describe the right person as ***"The other** person ..."* without adding any additional location information. Our current system can only handle individual language instructions separately and does not have the capability of understanding the contextual information.

The main reason for this limitation is that our *MATCHING* dataset was built automatically without any manual annotation. Consequently, the scope of handling language expressions is limited to variances within the dataset. One feasible solution is to distill and augment the dataset with crowd workers. For example, further research may invite users to polish the automatically synthesized text. Moreover, the dataset can be made more compatible with human's linguistic habits by asking users to combine the simple automatically synthesized sentences into more sophisticated expressions.

Lack of Language Generality for Colorization. Another limitation is that our system still cannot handle cases where the input instruction includes arbitrary part-level information (e.g., "the wheels" of the car) or arbitrary colors (e.g., "blonde hair"). In the future, the part-level information can be annotated from the WordNet [Miller 1995] while the color information can be computed as a linear mixture of several known colors. Our current dataset was designed with a narrow scope of objects and colors to reduce the manual effort of collecting the training and testing data, whereas it results in low variances within the dataset. In the future, we aim to facilitate additional networks to recognize and segment the parts and learn reasonable blended colors for the un-annotated parts.

Colorization Artifacts. In general, there still exist some artifacts in our results, such as uncolored pixels (e.g., between the boy's legs in Fig. 19) and aliasing artifact (region around the hair in Fig. 20 (d)). These artifacts are mainly caused by the limitation of the data. Our colorization scheme requires region-based segmentation information while the SketchyScene dataset only provides the stroke-based segmentation information. We applied a simple *poly2mask* tool to post-process the segmented strokes to obtain region-based segmentation masks for foreground objects, which may cause inaccurate region-based segmentation results (e.g., the blank region between the boy's legs in Fig. 19 was treated as a part of the boy). To deal with this problem, we plan to augment the SketchyScene dataset with region-based segmentation annotations, which may empower the foreground segmentation network to be a fully end-to-end model for producing region-based segmentation masks.

ACKNOWLEDGMENTS

We thank Xing Wu and other participants for helping us with data annotation. We also thank the anonymous reviewers for their insightful suggestions. This work was supported by the Fundamental Research Funds for the Central Universities, Guangzhou Science

Technology and Innovation Commission (GZSTI16EG14/201704030079). This work was also partially supported by a gift from Adobe, and grants from the RGC of HKSAR (Project No. CityU 11212119), City University of Hong Kong (Project No. 7005176) and the Centre for Applied Computing and Interactive Media (ACIM) of School of Creative Media, CityU.

REFERENCES

Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. 2017. SegNet: a Deep Convolutional Encoder-Decoder Architecture for Image Segmentation. *IEEE Transactions on Pattern Analysis & Machine Intelligence* 12 (2017), 2481–2495.

Hyojin Bahng, Seungjoo Yoo, Wonwoong Cho, David Keetae Park, Ziming Wu, Xiaojuan Ma, and Jaegul Choo. 2018. Coloring with words: Guiding image colorization through text-based palette generation. In *ECCV*. 431–447.

Huiwen Chang, Ohad Fried, Yiming Liu, Stephen DiVerdi, and Adam Finkelstein. 2015. Palette-Based Photo Recoloring. *ACM Transactions on Graphics* 34, 4 (2015), 139.

Jianbo Chen, Yelong Shen, Jianfeng Gao, Jingjing Liu, and Xiaodong Liu. 2018b. Language-Based Image Editing With Recurrent Attentive Models. In *CVPR*. IEEE.

Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. 2018a. Deeplab: Semantic Image Segmentation With Deep Convolutional Nets, Atrous Convolution, and Fully Connected Crfs. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 40, 4 (2018), 834–848.

Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. 2018c. Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation. In *ECCV*.

Wengling Chen and James Hays. 2018. Sketchygan: Towards diverse and realistic sketch to image synthesis. In *CVPR*. 9416–9425.

Ming-Ming Cheng, Shuai Zheng, Wen-Yan Lin, Vibhav Vineet, Paul Sturgess, Nigel Crook, Niloy J Mitra, and Philip Torr. 2014. ImageSpirit: Verbal guided image parsing. *ACM Transactions on Graphics (TOG)* 34, 1 (2014), 3.

Yuanzheng Ci, Xinzhu Ma, Zhihui Wang, Haojie Li, and Zhongxuan Luo. 2018. User-Guided Deep Anime Line Art Colorization With Conditional Adversarial Networks. In *ACM Multimedia*. 1536–1544.

Faming Fang, Tingting Wang, Tieyong Zeng, and Guixu Zhang. 2019. A Superpixel-based Variational Model for Image Colorization. *IEEE Transactions on Visualization and Computer Graphics* (2019).

Chie Furusawa, Kazuyuki Hiroshiba, Keisuke Ogaki, and Yuri Odagiri. 2017. Comicolorization: semi-automatic manga colorization. In *SIGGRAPH Asia 2017 Technical Briefs*. ACM, 12.

Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. 2017. Mask r-cnn. In *ICCV*. 2961–2969.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *CVPR*. 770–778.

Mingming He, Dongdong Chen, Jing Liao, Pedro V Sander, and Lu Yuan. 2018. Deep Exemplar-Based Colorization. *ACM Transactions on Graphics* 37, 4 (2018), 47.

Ronghang Hu, Marcus Rohrbach, and Trevor Darrell. 2016a. Segmentation From Natural Language Expressions. In *ECCV*. Springer, 108–124.

Ronghang Hu, Huazhe Xu, Marcus Rohrbach, Jiashi Feng, Kate Saenko, and Trevor Darrell. 2016b. Natural Language Object Retrieval. In *CVPR*. 4555–4564.

Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. 2017. Image-to-image translation with conditional adversarial networks. In *CVPR*. 1125–1134.

Hyunhoon Jung, Hee Jae Kim, Seongeun So, Jinjoong Kim, and Changhoon Oh. 2019. TurtleTalk: An Educational Programming Game for Children with Voice User Interface. In *Extended Abstracts of the 2019 CHI Conference on Human Factors in Computing Systems (CHI EA '19)*.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882* (2014).

Siwei Lai, Liheng Xu, Kang Liu, and Jun Zhao. 2015. Recurrent convolutional neural networks for text classification. In *AAAI*.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. *arXiv preprint arXiv:1603.01360* (2016).

Gierad P Laput, Mira Dontcheva, Gregg Wilensky, Walter Chang, Aseem Agarwala, Jason Linder, and Eytan Adar. 2013. Pixeltone: A multimodal interface for image editing. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 2185–2194.

Jianan Li, Yunchao Wei, Xiaodan Liang, Fang Zhao, Jianshu Li, Tingfa Xu, and Jiashi Feng. 2017. Deep attribute-preserving metric learning for natural language object retrieval. In *ACM Multimedia*. 181–189.

Mengtian Li, Zhe Lin, Radomir Mech, Ersin Yumer, and Deva Ramanan. 2019. Photo-Sketching: Inferring Contour Drawings from Images. In *WACV*. IEEE, 1403–1412.

Ruiyu Li, Kai-Can Li, Yi-Chun Kuo, Michelle Shu, Xiaojuan Qi, Xiaoyong Shen, and Jiaya Jia. 2018. Referring Image Segmentation via Recurrent Refinement Networks. In *CVPR*. 5745–5753.

Chenxi Liu, Zhe Lin, Xiaohui Shen, Jimei Yang, Xin Lu, and Alan Yuille. 2017a. Recurrent Multimodal Interaction for Referring Image Segmentation. In *ICCV*. IEEE.

Yifan Liu, Zengchang Qin, Zhenbo Luo, and Hua Wang. 2017b. Auto-Painter: Cartoon Image Generation From Sketch by Using Conditional Generative Adversarial Networks. *ArXiv Preprint ArXiv:1705.01908* (2017).

Jonathan Long, Evan Shelhamer, and Trevor Darrell. 2015. Fully convolutional networks for semantic segmentation. In *CVPR*. 3431–3440.

Silvia Lovato and Anne Marie Piper. 2015. "Siri, is This You?": Understanding Young Children's Interactions with Voice Input Systems. In *Proceedings of the 14th International Conference on Interaction Design and Children (IDC '15)*. 335–338.

Jiasen Lu, Jianwei Yang, Dhruv Batra, and Devi Parikh. 2016. Hierarchical Question-Image Co-Attention for Visual Question Answering. In *NIPS*. 289–297.

Junhua Mao, Jonathan Huang, Alexander Toshev, Oana Camburu, Alan L. Yuille, and Kevin Murphy. 2016. Generation and Comprehension of Unambiguous Object Descriptions. In *CVPR*. 11–20.

George A Miller. 1995. WordNet: a lexical database for English. *Commun. ACM* 38, 11 (1995), 39–41.

Will Monroe, Noah D. Goodman, and Christopher Potts. 2016. Learning to Generate Compositional Color Descriptions. In *EMNLP*.

Will Monroe, Robert XD Hawkins, Noah D Goodman, and Christopher Potts. 2017. Colors in context: A pragmatic neural model for grounded language understanding. *Transactions of the Association for Computational Linguistics* 5 (2017), 325–338.

Taesung Park, Ming-Yu Liu, Ting-Chun Wang, and Jun-Yan Zhu. 2019. Semantic image synthesis with spatially-adaptive normalization. In *CVPR*. 2337–2346.

Yingge Qu, Tien-Tsin Wong, and Pheng-Ann Heng. 2006. Manga colorization. In *ACM Transactions on Graphics (TOG)*, Vol. 25. 1214–1220.

Hayes Raffle, Cati Vaucelle, Ruibing Wang, and Hiroshi Ishii. 2007. Jabberstamp: Embedding Sound and Voice in Traditional Drawings. In *Proceedings of the 6th International Conference on Interaction Design and Children (IDC '07)*. 137–144.

Patsorn Sangkloy, Nathan Burnell, Cusuh Ham, and James Hays. 2016. The Sketchy Database: Learning to Retrieve Badly Drawn Bunnies. *ACM Transactions on Graphics (proceedings of SIGGRAPH)* (2016).

Patsorn Sangkloy, Jingwan Lu, Chen Fang, Fisher Yu, and James Hays. 2017. Scribbler: Controlling deep image synthesis with sketch and color. In *CVPR*. 5400–5409.

Hengcan Shi, Hongliang Li, Fanman Meng, and Qingbo Wu. 2018. Key-Word-Aware Network for Referring Expression Image Segmentation. In *ECCV*. 38–54.

Dong Wang, Changqing Zou, Guiqing Li, Chengying Gao, Zhuo Su, and Ping Tan. 2017. $\mathscr{L}0$ Gradient-Preserving Color Transfer. *Comput. Graph. Forum* 36, 7 (2017), 93–103.

Holger Winnemöller. 2011. Xdog: advanced image stylization with extended difference-of-gaussians. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Non-Photorealistic Animation and Rendering*. 147–156.

Chufeng Xiao, Chu Han, Zhuming Zhang, Jing Qin, Tien-Tsin Wong, Guoqiang Han, and Shengfeng He. 2019a. Example-Based Colourization Via Dense Encoding Pyramids. In *Computer Graphics Forum*. Wiley Online Library.

Yi Xiao, Peiyao Zhou, Yan Zheng, and Chi-Sing Leung. 2019b. Interactive Deep Colorization Using Simultaneous Global and Local Inputs. In *ICASSP*. IEEE, 1887–1891.

Tao Xu, Pengchuan Zhang, Qiuyuan Huang, Han Zhang, Zhe Gan, Xiaolei Huang, and Xiaodong He. 2018. Attngan: Fine-grained text to image generation with attentional generative adversarial networks. In *CVPR*. 1316–1324.

Xinchen Yan, Jimei Yang, Kihyuk Sohn, and Honglak Lee. 2016. Attribute2image: Conditional image generation from visual attributes. In *ECCV*. Springer, 776–791.

Taizan Yonetsuji. 2017. Paints Chainer. https://github.com/pfnet/PaintsChainer. (2017).

Dongfei Yu, Jianlong Fu, Tao Mei, and Yong Rui. 2017. Multi-level Attention Networks for Visual Question Answering. In *CVPR*.

Han Zhang, Tao Xu, Hongsheng Li, Shaoting Zhang, Xiaogang Wang, Xiaolei Huang, and Dimitris N Metaxas. 2017a. Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks. In *ICCV*. 5907–5915.

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

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 Transactions on Graphics (TOG)* 9, 4 (2017).

Changqing Zou, Qian Yu, Ruofei Du, Haoran Mo, Yi-Zhe Song, Tao Xiang, Chengying Gao, Baoquan Chen, and Hao Zhang. 2018. SketchyScene: Richly-Annotated Scene Sketches. In *ECCV*. Springer, 438–454.