iColoriT: Towards Propagating Local Hints to the Right Region in Interactive Colorization by Leveraging Vision Transformer

Jooyeol Yun*,   Sanghyeon Lee*,   Minho Park*,   Jaegul Choo

Korea Advanced Institute of Science and Technology (KAIST)

Daejeon, Korea

{blizzard072, shlee6825, m.park, jchoo}@kaist.ac.kr

Abstract

Point-interactive image colorization aims to colorize grayscale images when a user provides the colors for specific locations. It is essential for point-interactive colorization methods to appropriately propagate user-provided colors (i.e., user hints) in the entire image to obtain a reasonably colorized image with minimal user effort. However, existing approaches often produce partially colorized results due to the inefficient design of stacking convolutional layers to propagate hints to distant relevant regions. To address this problem, we present iColoriT, a novel point-interactive colorization Vision Transformer capable of propagating user hints to relevant regions, leveraging the global receptive field of Transformers. The self-attention mechanism of Transformers enables iColoriT to selectively colorize relevant regions with only a few local hints. Our approach colorizes images in real-time by utilizing pixel shuffling, an efficient upsampling technique that replaces the decoder architecture. Also, in order to mitigate the artifacts caused by pixel shuffling with large upsampling ratios, we present the local stabilizing layer. Extensive quantitative and qualitative results demonstrate that our approach highly outperforms existing methods for point-interactive colorization, producing accurately colorized images with a user's minimal effort. Official codes are available at https://pmh9960.github.io/research/iColoriT/.

1. Introduction

Unconditional image colorization [11, 12, 30, 32, 39, 41] has shown remarkable achievement in restoring the vibrance of grayscale photographs or films in a fully-automatic manner. Interactive colorization methods [7, 15, 36, 37, 40, 43] further extend the task to allow users to generate colorized images with specific color conditions. These approaches can dramatically reduce the user effort for producing specific colorized images. It can also serve as an effective way of editing photos by re-coloring existing images to have a new color theme. Among different types of inter-
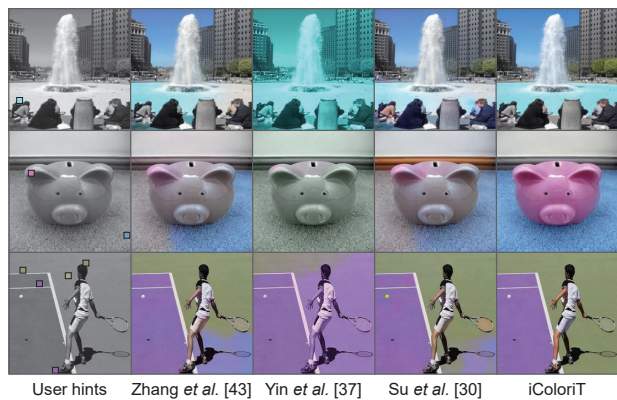
* indicates equal contribution.

Figure 1. Example results of various point-interactive colorization approaches. Previous approaches often produce partially colorized results even where the grayscale values are persistent (*e.g.*, water, floor, and grass), which indicates that the user hints did not properly propagate to the relevant regions.

actions provided by users (*e.g.*, a reference image or a color palette), point- or scribble-based interactions [15, 37, 43] are designed to progressively colorize images when a user provides the colors at specific point locations.

Practical point-interactive colorization methods assist the user to produce a colorized image with minimal user interaction. Thus, accurately estimating the regions relevant to the user hint can be beneficial for reducing the amount of user interactions. For example, using hand-crafted filters [15, 37] to determine the region a user hint should fill in was an early approach for colorizing simple patterns within the image. Recently, Zhang *et al.* [43] proposed a learning-based model trained on a large-scale dataset [26] which produces colorized images with a simple U-Net architecture. However, existing methods tend to suffer from partially colorized results even in obvious regions where the grayscale values are persistent, as seen in Figure 1. This is due to the inefficient design of stacking convolutional layers in order to propagate hints to distant relevant regions. In other words, propagating hints to large semantic regions can only be done in the deep layers, which makes colorizing

larger semantic regions more challenging than colorizing smaller regions. To overcome this hurdle, we leverage the global receptive field of self-attention layers [31] in Vision Transformers [4], enabling the model to selectively propagate user hints to relevant regions at each single layer.

Learning how to propagate user hints to other regions aligns well with the self-attention mechanism. Specifically, directly computing the similarities of features from all spatial locations (*i.e.*, the similarity matrix) can be viewed as deciding where the hint colors should propagate in the entire image. Thus, in this work, we present iColoriT, a novel point-interactive colorization framework utilizing a modified Vision Transformer for colorizing grayscale images. To the best of our knowledge, this is the first work to employ a Vision Transformer for point-interactive colorization.

Furthermore, promptly displaying the results for a newly provided user hint is essential for assisting users to progressively colorize images without delay. For this reason, we generate color images by leveraging the efficient pixel shuffling operation [27], an upsampling technique that reshapes the output channel dimension into a spatial resolution. Through the light-weight pixel shuffling operation, we are able to discard the conventional decoder architecture and offer a faster inference speed compared to existing baselines. Despite its efficiency, pixel shuffling with large upsampling ratios tends to generate unrealistic images with missing details and notable boundaries as seen in Figure 2. Therefore, we present the local stabilizing layer, which restricts the receptive field of the last layer, to mitigate the artifacts caused by pixel shuffling. Our contributions are as follows:

- We are the first work to utilize a Vision Transformer for point-interactive colorization enabling users to selectively colorize relevant regions.

- We achieve real-time colorization of images by effectively upsampling images with minimal cost, leveraging the pixel shuffling and the local stabilizing layer.

- We provide quantitative and qualitative results demonstrating that iColoriT highly outperforms existing state-of-the-art baselines and generates reasonable results with fewer user interactions.

2. Related Work

Interactive Colorization Learning-based methods for image colorization [11–13, 13, 30, 32, 39, 41, 44] have proposed fully-automated colorization methods, which generate reasonable color images without the need of any user intervention. Interactive colorization methods [7, 15–17, 21, 34, 36–38, 40, 43] are designed to colorize images given a user's condition which conveys color-related information. A widely-studied condition type for interactive colorization

Figure 2. Images generated with large upsample ratios [5] tends to suffer from evident borders between image patches.

methods are reference images [7, 16, 17, 21, 34, 36, 38, 40], which are already-colored exemplar images. Using reference images can be convenient since the user can provide the overall color tones with a single image. However, it is difficult for the user to further edit specific regions in the colorized image since a new reference image is likely to produce a different colorization result.

Point-interactive Colorization Point-interactive colorization models [15, 37, 43] allow the user to progressively colorize images by specifying colors (*i.e.*, user hints) at different point locations in the input grayscale image. Since commonly used point sizes for specifying the spatial locations range from 2×2 to 7×7 pixels, the user hints only cover a small portion of the entire image. Thus, a point-interactive colorization model is required to propagate user hints to the entire image in order to produce a reasonable result with minimal user interaction. Early approaches [15, 37] utilized hand-crafted image filters to determine the propagation region of each hint by detecting simple patterns. The colors of the user hints are then propagated within each region using optimization techniques. Recently, Zhang *et al.* [43] proposed a learning-based method by extending an existing unconditional colorization model [41] to produce color images given a grayscale image and user hints. Although these methods use user hints as a condition for generating color images, common failure cases presented in Figure 1 indicate that the models often propagate hints incompletely. Stacking convolutional layers to propagate user hints indicates that propagating hints to distant relevant regions can only be done in the deeper layers, which makes colorizing larger semantic regions more challenging than nearby regions. Thus, we utilize the self-attention layer to enable user hints to propagate to any relevant regions at all layers.

Image Colorization with Transformers Unlike the widely-used convolution-based approach for image synthesis, recent studies [5, 12, 14, 35] made efforts to synthesize images by only utilizing the Transformer architecture. Colorization Transformer (ColTran) [12] proposes an autoregressive model for unconditional colorization which uses the Transformer decoder architecture [31] in order to generate diverse colorization results. Despite its outstanding performance for unconditional colorization, the excessively slow inference speed of autoregressive models hinders its application to user-interactive scenarios. Specifically, it takes 3.5-5 minutes to colorize a batch of 20 images
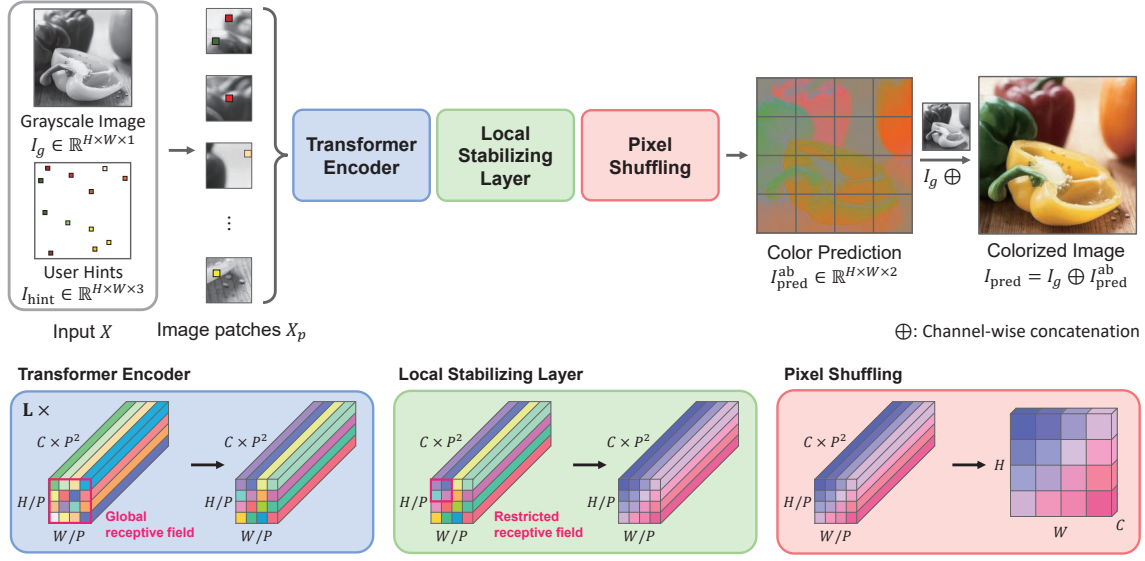
Figure 3. The overall workflow of iColoriT. We first obtain input X by concatenating the grayscale image I_g and the user hint I_{hint} containing color conditions. The input is reshaped into input patches for the Transformer encoder. The output features from the Transformer encoder are passed through the local stabilizing layer and the pixel shuffling layer to obtain the final colors $I_{\text{pred}}^{\text{ab}}$. $I_{\text{pred}}^{\text{ab}}$ is then concatenated with I_g to produce the colorized image.

of size 64×64 images even with a P100 GPU. In this work, we leverage the Transformer *encoder* to generate the colors of a grayscale image. The multi-head attention of the Transformer encoder enables our approach to generate color images with a single forward pass which reduces the inference time of our model compared to autoregressive colorization.

Upsampling via Pixel Shuffling Pixel shuffling [27] is an upsampling operation that rearranges a $(H, W, C \times P^2)$ sized feature map into a shape of $(H \times P, W \times P, C)$ where each channel in the original feature map is reshaped into a $P \times P$ image patch. This can be viewed as *upsampling via reshaping*, and is often used in super-resolution approaches to effectively upsample an image with minimal computational overhead. A known issue [5] with pixel shuffling with larger upsampling ratios ($P > 8$) was that output images tend to contain evident borders between image patches as seen in Figure 2. This is due to upsampling different image patches from different locations in the feature map. To overcome this hurdle, we present a local stabilizing layer, which promotes neighboring image patches to have coherent colors, allowing iColoriT to effectively upsample images to higher resolutions (*i.e.*, 224×224) without such artifacts.

3. Proposed Method

3.1. Preliminaries

We first prepare the grayscale image $I_g \in \mathbb{R}^{H \times W \times 1}$ and the simulated user hints $I_{\text{hint}} \in \mathbb{R}^{H \times W \times 3}$ to be used as our training sample. A grayscale image I_g can be acquired from large-scale datasets by converting the color

space from RGB to CIELab [28] and taking the L or lightness value. Similarly, the color condition I_{hint} provided by the user can be expressed with the remaining a, b channel values $\tilde{I}_{\text{hint}} \in \mathbb{R}^{H \times W \times 2}$ by filling the a,b channel values of all non-hint regions with 0. The user hint $I_{\text{hint}} \in \mathbb{R}^{H \times W \times 3}$ is constructed by adding a third channel to $\tilde{I}_{\text{hint}}$ that marks hint regions with 1 and non-hint regions with 0.

During training, we simulate the user hints by determining the hint location and the color of the hint. We sample hint locations from a uniform distribution since a user may provide hints anywhere in the image. Once the hint location is decided, the color of the user hint is obtained by calculating the average color values for each channel within the hint region since a user is expected to provide a single color for a single hint location. Finally, given the grayscale image $I_g \in \mathbb{R}^{H \times W \times 1}$ and the simulated user hints $I_{\text{hint}} \in \mathbb{R}^{H \times W \times 3}$, we obtain our input $X \in \mathbb{R}^{H \times W \times 4}$ by

$$X = I_g \oplus I_{\text{hint}},$$

where $\oplus$ is the channel-wise concatenation.

3.2. Propagating User Hints with Transformers

We utilize the Vision Transformer [4] to achieve a global receptive field for propagating user hints across the image as shown in Figure 3. We first reshape our input $X \in \mathbb{R}^{H \times W \times 4}$ into a sequence of tokens $X_p \in \mathbb{R}^{N \times (P^2 \times 4)}$, where H, W are the height and width of the original image, P is the patch size, and $N = HW/P^2$ is the number of input tokens (*i.e.*, sequence length). Thus, a $P \times P \times 4$ size image patch from the original input X is used as a sin-

gle input token. These sequence of input tokens are passed through the Transformer encoder, which computes the input as,

$$z_0 = X_p + E_{pos}, \quad E_{pos} \in \mathbb{R}^{N \times d} \quad (1)$$

$$z_l' = \text{MSA}(\text{LN}(z_{l-1})) + z_{l-1}, \quad (2)$$

$$z_l = \text{MLP}(\text{LN}(z_l')) + z_l', \quad (3)$$

$$y_p = \text{LN}(z_L), \quad (4)$$

where E_{pos} denotes the sinusoidal positional encoding [4], $\text{MSA}(\cdot)$ indicates the multi-head self-attention [31], $\text{LN}(\cdot)$ indicates the layer normalization [2], d denotes the hidden dimension, l denotes the layer number, and $y_p \in \mathbb{R}^{N \times d}$ denotes the output of the Transformer encoder. Since self-attention does not utilize any position-related information, we add positional encoding E_{pos} to the input and relative positional bias [8, 9, 18, 25] in the attention layer. Thus, the attention layer is computed as,

$$\text{Attention}(Q, K, V) = \text{softmax}(QK^T/\sqrt{d} + B)V, \quad (5)$$

where $Q, K, V \in \mathbb{R}^{N \times d}$ are the query, key and value matrices, $B \in \mathbb{R}^{N \times N}$ is the relative positional bias. The colors of the user hints are able to propagate to any spatial location at all layers due to the global receptive field of the self-attention mechanism.

3.3. Pixel Shuffling and the Local Stabilizing Layer

The output features of the Transformer encoder $y_p \in \mathbb{R}^{N \times d}$ can be viewed as a feature map $y \in \mathbb{R}^{H/P \times W/P \times d}$ of the original image. The spatial resolution of the output feature map y is smaller than the resolution of the input image by a factor of P since image patches of size $P \times P$ consists of a single input token. Therefore, the output feature map y needs to be upsampled in order to obtain a full-resolution color image. While previous approaches [30, 43] leverage a decoder for upsampling, we utilize pixel shuffling [27] which is an upsampling technique rearranging a $(H/P, W/P, C \times P^2)$ feature map into a shape of (H, W, C) to obtain a full-resolution image.

However, as mentioned in Section 2, large upsampling ratios (*e.g.*, $P > 8$) may lead to images with visible artifacts along the image patch boundaries as seen in Figure 4. Thus, in order to promote reasonable generation of colors, we propose a local stabilizing layer, which restricts the model to generate colors utilizing neighboring features, and place the layer before pixel shuffling. We provide experiments in Section 4.2 with various design choices for the local stabilizing layer (*e.g.*, linear, convolutional layer, and local attention) and select a simple yet effective convolutional layer as our final model. To sum up, our upsampling process can be written as,

$$I_{\text{pred}}^{\text{ab}} = \mathcal{PS}(\text{LS}(y)), \quad (6)$$

iColoriT w/o Local Stabilizing Layer   iColoriT

Figure 4. Example images of inconsistent colorization results observed in images produced by without the local stabilizing layer.

where $\mathcal{PS}(\cdot)$ is the pixel shuffling operation, $\text{LS}(\cdot)$ is the local stabilizing layer, and $I_{\text{pred}}^{\text{ab}} \in \mathbb{R}^{H \times W \times 2}$ is the ab color channel outputs. The predicted color image $I_{\text{pred}} \in \mathbb{R}^{H \times W \times 3}$ is obtained by

$$I_{\text{pred}} = I_g \oplus I_{\text{pred}}^{\text{ab}},$$

which is the concatenation of the given grayscale input I_g (L channel) and $I_{\text{pred}}^{\text{ab}}$ (ab channel). Through pixel shuffling and the local stabilizing layer, we can effectively obtain a full-resolution color image *without* an additional decoder, allowing real-time colorization for the user (Section 4.1).

3.4. Objective Function

We train our model with the Huber loss [10] between the predicted image and the original color image in the CIELab color space,

$$\begin{aligned} L_{recon} = &\frac{1}{2}(I_{\text{pred}} - I_{GT})^2 \mathbb{1}_{|I_{\text{pred}} - I_{GT}| < 1} \\ &+ (|I_{\text{pred}} - I_{GT}| - \frac{1}{2})\mathbb{1}_{|I_{\text{pred}} - I_{GT}| \geq 1}. \end{aligned} \quad (7)$$

4. Experiments

Implementation Details We follow the configurations of ViT-B [4] for the Transformer encoder blocks. For the local stabilizing layer, we use a single layer with a receptive field of 3. We experiment with two types of layers (Section 4.2), the local attention and the convolutional layer, and use the simple yet effective convolutional layer as the default local stabilizing layer. For training, we resize images to a 224×224 resolution and use a patch size of $P = 16$ which also becomes the upsampling ratio. Thus, the sequence length N is 196 and the last output dimension d is 512. We sample hint locations uniformly across the image and sample the number of hints from a uniform distribution $\mathcal{U}(0, 128)$. We provide experiments on different model sizes, patch sizes, the local stabilizing layer, and the number of hints in Section 4.2 and the supplementary material.
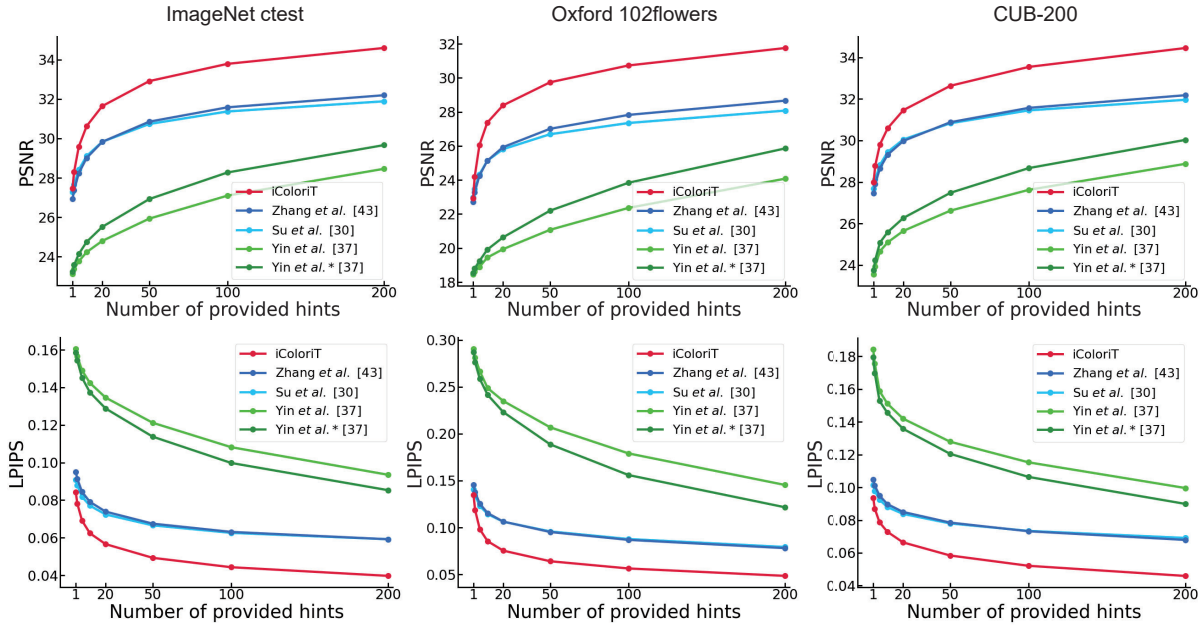
Figure 5. Average PSNR and LPIPS of the test images according to the number of provided hints. Hint locations are sampled from a uniform distribution and 2×2 hints are revealed to the model. Yin et al. * [37] denotes the results evaluated with 2×2 hints and Yin et al. [37] denotes the results evaluated with 7×7 hints. iColoriT outperforms existing approaches by a large margin as the number of provided hints increases.

We use the AdamW optimizer [20] with a learning rate of 0.0005 managed by the cosine annealing scheduler [19]. The model is trained for 2.5M iterations with a batch size of 512. The codes for iColoriT implemented with the Pytorch library [23] will be available.

Datasets For training, we use the ImageNet 2012 train split [26] which consists of 1,281,167 images. We do not use the classification labels during training since our model is trained in a self-supervised manner. We evaluate our method on three datasets from different domains, all of which are colorful validation datasets suitable for evaluating colorization approaches. Note that we do not additionally finetune the model for each validation dataset. The ImageNet ctest [13] is a subset of the ImageNet validation split used as a standard benchmark for evaluating colorization models. ImageNet ctest excludes any grayscale image from ImageNet and consists of 10,000 color images. We also evaluate on the Oxford 102flowers dataset [22] and the CUB-200 dataset [33] which provide 1,020 colorful flower images from 102 categories and 3,033 samples of bird images from 200 different species, respectively.

Baselines We compare the performance of iColoriT with existing interactive colorization methods [37, 43]. We also extend a recent unconditional colorization model by Su et al. [30], which utilizes an off-the-shelf object detector [6] to individually color multiple instances, to a point-interactive colorization model. Since the model proposed by Su et al. [30] employs the same model architecture and objec-

tive function as the point-interactive colorization model by Zhang et al. [43], we are able to effortlessly extend the approach to a point-interactive colorization method by conditioning the model with user hints in the same manner. The extended model is trained under the configurations provided by Zhang et al. [43] and Su et al. [30] using ImageNet [26]. Note that although the model proposed by Su et al. [30] is trained with the ImageNet [26] dataset, this approach is assisted by an off-the-shelf object detector pre-trained on a large-scale object detection dataset [3]. All baselines are trained and evaluated with the publicly available official codes.

4.1. Comparison with Existing Approaches

Quantitative Evaluation of iColoriT We plot the average peak signal-to-noise ratio (PSNR) and the learned perceptual image patch similarity (LPIPS) [42] of the test images according to the number of provided hints in Figure 5. For evaluating the point-interactive colorization models, we simulate user hints with the ground-truth colors from the image, considering a situation where the user intends to colorize the grayscale image into the original color image. User hints are simulated by randomly selecting hint locations from a uniform distribution. The hint sizes are set to 2×2 and the hint color is given as the average color within each hint region in the original color image following the protocol of Zhang et al. [43]. We empirically find that smaller hint sizes are usually beneficial for both the colorization model and the user in terms of receiving and

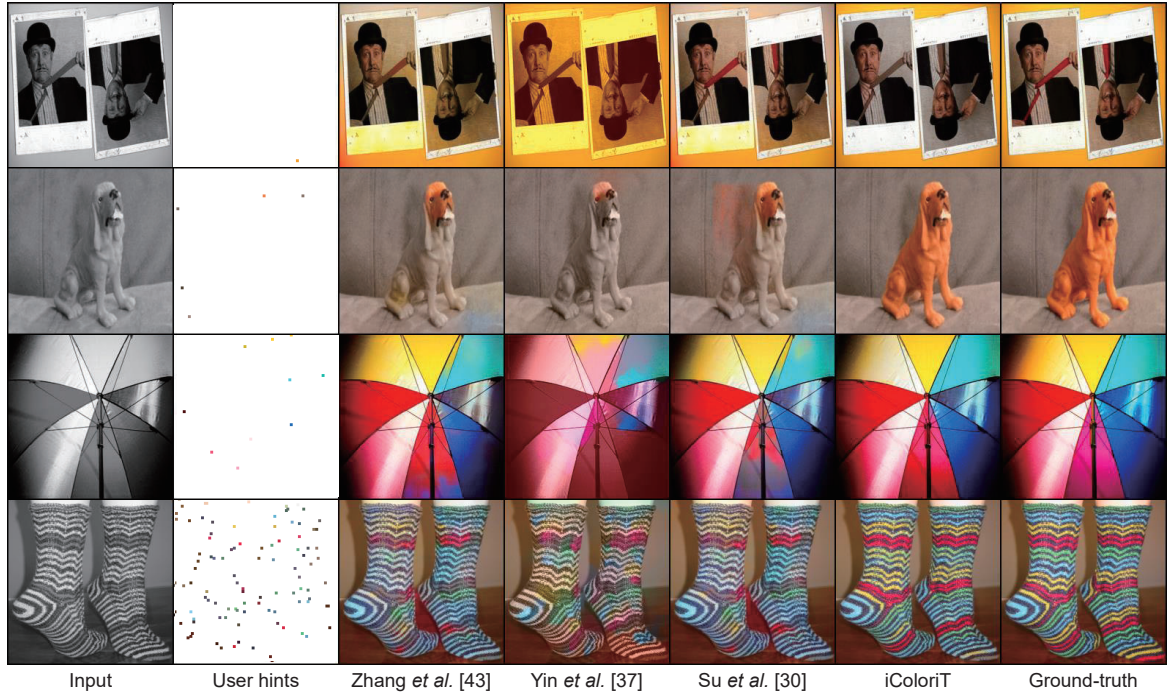| Input | User hints | Zhang *et al*. [43] | Yin *et al*. [37] | Su *et al*. [30] | iColoriT | Ground-truth |

Figure 6. Qualitative results of point-interactive colorization methods given 1, 5, 10, and 100 user hints. iColoriT is able to produce reasonable color images by appropriately propagating user hints.

giving accurate color conditions. However, the method proposed by Yin *et al*. [37] assumes that a user provides an abundant amount of user hints. Thus, we further evaluate this method by revealing larger hints of size 7×7 which is the result we report for all following evaluations.

We empirically find that methods proposed by Zhang *et al*. [43] and Su *et al*. [30] tend to arbitrarily colorize images without reflecting user hints. While this may be helpful for achieving a relatively higher initial PSNR when the arbitrarily colorized color is the ground-truth color, it hinders further control for the user to achieve a high PSNR in subsequent stages of colorization. As seen in Figure 5, iColoriT quickly reflects the user hints and aids the user to efficiently colorize grayscale images with minimal interaction. The PSNR in the early stages of colorization notably increases with each additional hint. The results indicate that iColoriT highly outperforms existing baselines for generating colorized images a user specifically has in mind.

Qualitative Results of iColoriT We provide qualitative results produced by the baselines and iColoriT in Figure 6 when given an original grayscale image and the simulated user hints. iColoriT is able to produce realistic images that closely resemble the ground-truth image indicating that a user can colorize images as they please. Also, as seen in the colorized results in Figure 1 and Figure 6, iColoriT is capable of appropriately colorizing large areas even with a small number of user hints while other approaches leave most regions uncolored or incorrectly colored. iColoriT can also

Methods	PSNR@10	LPIPS@10
iColoriT-T	28.86	0.084
iColoriT-S	29.67	0.073
iColoriT	**30.63**	**0.062**

Table 1. Scalability of iColoriT to lightweight models. PSNR and LPIPS given 10 user hints (PSNR@10 and LPIPS@10) on the ImageNet ctest [13] are reported for each model.

colorize detailed regions when given a sufficient number of hints as shown in the last row of Figure 6.

iColoriT is also suitable for producing *diverse* colorized images when given various user hints as seen in Figure 7. Instead of the simulated user hints from the ground-truth image, we provide multiple sets of hand-picked user hints to colorize a single grayscale image. We fix the hint locations for an image and alter the user-provided colors to observe the colorized results. iColoriT can produce various realistic colorization results that reflect the intention of the user. We provide uncurated qualitative results and a demo video in the supplementary material. Also, we will release the iColoriT demo including the graphical user interface, providing a powerful tool for image colorization.

Scaling to Lightweight Models iColoriT can easily scale to smaller models and still achieve high performance. We train iColoriT in smaller scales using the configurations of the ViT-S and the ViT-Ti [29] for our Transformer encoder. We report the PSNR and the LPIPS given 10 hints
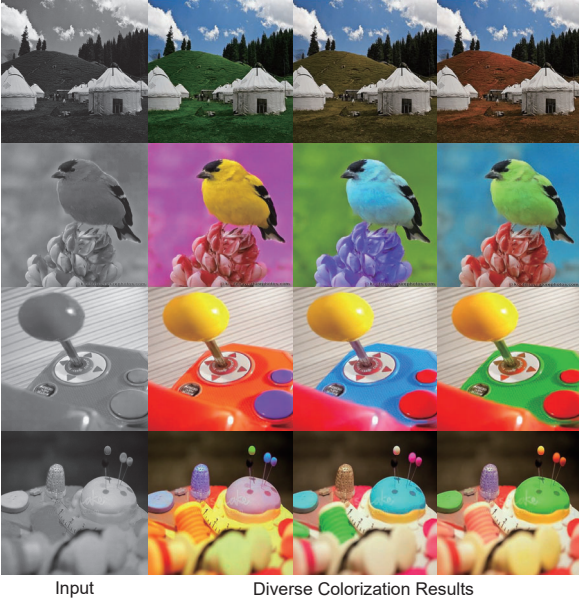
Figure 7. Images colorized with different colors provided by the user. The images from the ImageNet ctest [13] are colored by hand-picking hint locations and changing the hint colors.

(PNSR@10 and LPIPS@10) for ImageNet ctest and compare them against other models in Table 1. We were able to train iColoriT-S and iColoriT-T with only a slight performance drop and still maintain a high performance. We believe that the Transformer architecture and the self-attention mechanism are central for propagating hints to larger semantic regions, achieving a high PSNR even in small-scale models.

Real-time Inference The inference speed (*i.e.*, latency) of point-interactive models is important for providing a satisfying user experience. Thus, we measure the time required for a single forward pass and compare it with the latency of baseline models in Table 2. We report the speed on both CPU and GPU using a commercial AMD Ryzen 5 PRO 4650G and a single NVIDIA RTX 3090. We also provide the number of floating-point operations (FLOPs) and the number of parameters required for each model. We were not able to measure GPU latency, FLOPs, and the number of parameters for Yin *et al*. [37] since the method is not a learning-based model. The model proposed by Su *et al*. [30] operates in two stages, an initial object detection stage and an instance-wise colorization stage. We only report the latency for the second stage which still exhibits a slow inference speed since the colorization model needs to color multiple objects individually. Due to the efficient pixel shuffling for upsampling images, iColoriT enjoys a short latency of 540ms and 14ms on a CPU and GPU device respectively, providing real-time colorization results for the user. iColoriT-T and iColoriT-S show an exception-

Methods	CPU Latency	GPU Latency	GFLOPs
Zhang *et al*. [43]	881ms	24ms	58.04
Yin *et al*. [37]	15,248ms	-	-
Su *et al*. [30]	1,389ms	45ms	123.48
iColoriT-T	**177ms**	**13ms**	**1.43**
iColoriT-S	253ms	14ms	4.95
iColoriT	540ms	14ms	18.22

Table 2. Inference speed of iColoriT and each baseline model. We provide the latency of each model in a CPU device and a GPU device along with the computational cost measured in FLOPs and number of parameters.

ally fast inference speed on a CPU-only device (*i.e.*, 177ms and 253ms, respectively), which makes the model an appealing option when considering applications to real-world scenarios where accelerators may not be available.

4.2. Ablation Study

Designing the Local Stabilizing Layer We provide an ablation study on the local stabilizing layer by replacing it with different operations such as the linear layer and the local self-attention layer [24]. Using a linear layer can be viewed as eliminating the local stabilizing layer since a linear layer does not utilize neighboring features for generating the final output. In order to quantify the inconsistent color generation among image patches seen in Figure 4, we measure the mean squared error (MSE) for each image patch and report the variance of the errors within an image. We denote this measure the patch error variance (PEV). A high PEV implies that the model has varying accuracy depending on the image patch. The local stabilizing layer resolves this issue in a simple yet effective manner by predicting the ab channel values of an image patch from neighboring output features as illustrated in Figure 3. We also measure the PSNR near the image patch boundaries (*i.e.*, one pixel from the patch borders) to observe the accuracy in the regions containing inconsistent color generation. As seen in Table 3, adding an operation with a limited receptive field (*i.e.*, convolution and local self-attention) lowers the PEV and increases the PSNR along the patch boundaries, indicating that the model generates colors with consistent accuracy across the image. The convolutional layer serves as a simple yet effective approach for reducing artifacts caused by pixel shuffling and generating realistic colorized images.
Changing the Upsampling Ratio We experiment on various patch sizes P (*i.e.*, $P = 8, 16$, and 32), which also becomes the upscaling ratio for pixel shuffling. While smaller patch sizes may allow fine-grained calculation of the similarity matrix, the computational cost escalates *biquadratically*, since the computational complexity for the self-attention follows $\mathcal{O}(N^2)$ and $N = HW/P^2$ is the sequence length. Thus, we were not able to train our base

Methods	PSNR@10	B-PSNR@10	PEV↓
Linear	28.78	28.71	39.39
Local Attention	28.85	28.77	38.82
Convolution	**28.86**	**28.80**	**38.81**

Table 3. Ablation study on the local stabilizing layer. PSNR@10, PSNR along the boundary (B-PSNR@10), and PEV on the ImageNet ctest [13] are reported for each model. All models are trained with the iColoriT-T configuration.

Patch Size	PSNR@10	CPU Latency
8×8	29.17(+0.31)	373ms(+196ms)
32×32	28.32(−0.54)	147ms(−30ms)
16×16	28.86	177ms

Table 4. iColoriT different upsampling ratios. PSNR@10, LPIPS@10, and CPU latency are reported for each model on the ImageNet ctest [13] test set. All models are trained with the iColoriT-T configuration.

model with a smaller patch size due to the prohibitive computational overhead. Instead, we compare the results on the smaller iColoriT-T model and report the average PSNR@10 and CPU latency on Table 4. While using a smaller patch size may be beneficial for achieving a higher PSNR, the increased computational cost hinders scaling to larger models for an additional performance gain and increases the CPU latency. We choose a patch size of 16×16 since it can obtain both a short latency and a high PSNR while also being scalable to larger models (*i.e.*, iColoriT-S and iColoriT).

4.3. Visualizing the Internal Representation

We further provide analysis on the self-attention mechanism to examine how our model is propagating user hints to other regions. We use the attention rollout method [1] to interpret the attention weights from the Transformer encoder for specific spatial locations. We visualize the attention maps for the input tokens which contain a user hint in Figure 8. Attention maps for hint locations can be directly interpreted as how the hint is propagating to other locations since tokens with high similarities are likely to be colorized with similar color as the color of the user hint. The self-attention mechanism enables iColoriT to selectively colorize relevant locations, even for regions with spatially complicated structures. These visualization aligns well with our qualitative and quantitative results demonstrating that iColoriT can effectively aid users to colorize images with minimal interaction.

5. Conclusion and Limitations

In this paper, we present iColoriT, a novel real-time point-interactive colorization framework capable of selectively propagating colors of the user hints to relevant regions. Through the Transformer encoder, pixel shuffling and the local stabilizing layer, iColoriT highly outperforms ex-
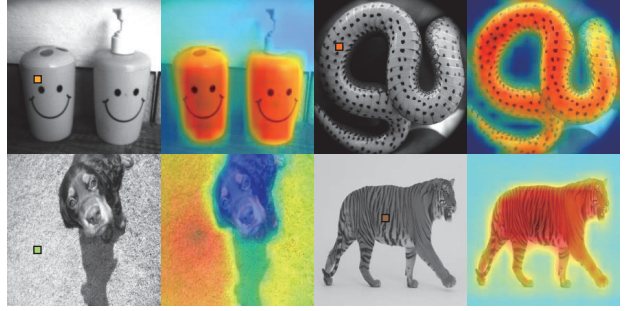

Figure 8. Visualization of the self-attention mechanism employing the attention rollout [1] method. iColoriT appropriately attends the user hint to relevant locations even for complex structures.
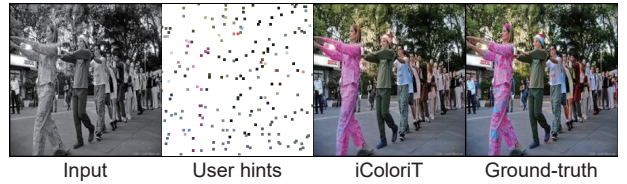

Input	User hints	iColoriT	Ground-truth

Figure 9. A common failure case for point-interactive colorization models in detailed regions.

isting baselines, being able colorize images with minimal user interaction. Also, qualitative results indicate that iColoriT can generate diverse and realistic results when given various user hints. We justify our novel design through extensive experiments and ablation studies.

Although iColoriT shows its strength even in detailed regions as shown in both quantitative and qualitative results, iColoriT may not be able to colorize small objects or distinguish close objects with the same grayscale intensity, since it does not leverage any semantic labels. This is a common drawback of point-interactive colorization approaches as seen in Figure 9 since models are trained in a self-supervised manner. Directly utilizing segmentation labels for training a point-interactive colorization model can be a promising future work. Nonetheless, we believe that the iColoriT is a practical application for real-world scenarios, effectively assisting the user to colorize images.

Acknowledgement

References

[1] Samira Abnar and Willem Zuidema. Quantifying attention flow in transformers. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4190–4197, 2020.

[2] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

[3] Holger Caesar, Jasper Uijlings, and Vittorio Ferrari. Cocostuff: Thing and stuff classes in context. In *Proc. of the IEEE conference on computer vision and pattern recognition (CVPR)*, 2018.

[4] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2020.

[5] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. *arXiv preprint arXiv:2111.06377*, 2021.

[6] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017.

[7] Mingming He, Dongdong Chen, Jing Liao, Pedro V Sander, and Lu Yuan. Deep exemplar-based colorization. *ACM Transactions on Graphics (TOG)*, 37(4):1–16, 2018.

[8] Han Hu, Jiayuan Gu, Zheng Zhang, Jifeng Dai, and Yichen Wei. Relation networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3588–3597, 2018.

[9] Han Hu, Zheng Zhang, Zhenda Xie, and Stephen Lin. Local relation networks for image recognition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3464–3473, 2019.

[10] Peter J Huber. Robust estimation of a location parameter. In *Breakthroughs in statistics*, pages 492–518. Springer, 1992.

[11] Satoshi Iizuka, Edgar Simo-Serra, and Hiroshi Ishikawa. Let there be color!: Joint end-to-end learning of global and local image priors for automatic image colorization with simultaneous classification. *Proc. the ACM Transactions on Graphics (ToG)*, 35, 2016.

[12] Manoj Kumar, Dirk Weissenborn, and Nal Kalchbrenner. Colorization transformer. In *ICLR*, 2020.

[13] Gustav Larsson, Michael Maire, and Gregory Shakhnarovich. Learning representations for automatic colorization. In *European Conference on Computer Vision (ECCV)*, 2016.

[14] Kwonjoon Lee, Huiwen Chang, Lu Jiang, Han Zhang, Zhuowen Tu, and Ce Liu. Vitgan: Training gans with vision transformers. In *International Conference on Learning Representations*, 2021.

[15] Anat Levin, Dani Lischinski, and Yair Weiss. Colorization using optimization. *Proc. the ACM Transactions on Graphics (ToG)*, 23:689–694, 2004.

[16] Bo Li, Yu-Kun Lai, Matthew John, and Paul L Rosin. Automatic example-based image colorization using location-aware cross-scale matching. *IEEE Transactions on Image Processing*, 28(9):4606–4619, 2019.

[17] Haoxuan Li, Bin Sheng, Ping Li, Riaz Ali, and CL Philip Chen. Globally and locally semantic colorization via exemplar-based broad-gan. *IEEE Transactions on Image Processing*, 30:8526–8539, 2021.

[18] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *ICCV*, 2021.

[19] Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. In *International Conference on Learning Representations*, 2017.

[20] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2018.

[21] Peng Lu, Jinbei Yu, Xujun Peng, Zhaoran Zhao, and Xiaojie Wang. Gray2colornet: Transfer more colors from reference image. In *Proceedings of the 28th ACM International Conference on Multimedia*, pages 3210–3218, 2020.

[22] Maria-Elena Nilsback and Andrew Zisserman. Automated flower classification over a large number of classes. In *Indian Conference on Computer Vision, Graphics and Image Processing*, 2008.

[23] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Proc. the Advances in Neural Information Processing Systems (NeurIPS)*, 32:8026–8037, 2019.

[24] Jack Rae and Ali Razavi. Do transformers need deep long-range memory? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Online, July 2020. Association for Computational Linguistics.

[25] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21:1–67, 2020.

[26] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015.

[27] Wenzhe Shi, Jose Caballero, Ferenc Huszár, Johannes Totz, Andrew P Aitken, Rob Bishop, Daniel Rueckert, and Zehan Wang. Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1874–1883, 2016.

[28] Thomas Smith and John Guild. The cie colorimetric standards and their use. *Transactions of the optical society*, 33(3):73, 1931.

[29] Andreas Steiner, Alexander Kolesnikov, Xiaohua Zhai, Ross Wightman, Jakob Uszkoreit, and Lucas Beyer. How to train your vit? data, augmentation, and regularization in vision transformers. *arXiv preprint arXiv:2106.10270*, 2021.

[30] Jheng-Wei Su, Hung-Kuo Chu, and Jia-Bin Huang. Instance-aware image colorization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7968–7977, 2020.

[31] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[32] Patricia Vitoria, Lara Raad, and Coloma Ballester. Chromagan: Adversarial picture colorization with semantic class distribution. In *WACV*, 2020.

[33] P. Welinder, S. Branson, T. Mita, C. Wah, F. Schroff, S. Belongie, and P. Perona. Caltech-ucsd birds 200. Technical Report CNS-TR-2010-001, California Institute of Technology, 2010.

[34] Chufeng Xiao, Chu Han, Zhuming Zhang, Jing Qin, Tien-Tsin Wong, Guoqiang Han, and Shengfeng He. Example-based colourization via dense encoding pyramids. In *Computer Graphics Forum*, volume 39, pages 20–33. Wiley Online Library, 2020.

[35] Zhenda Xie, Zheng Zhang, Yue Cao, Yutong Lin, Jianmin Bao, Zhuliang Yao, Qi Dai, and Han Hu. Simmim: A simple framework for masked image modeling. *arXiv preprint arXiv:2111.09886*, 2021.

[36] Zhongyou Xu, Tingting Wang, Faming Fang, Yun Sheng, and Guixu Zhang. Stylization-based architecture for fast deep exemplar colorization. In *Proc. of the IEEE conference on computer vision and pattern recognition (CVPR)*, pages 9363–9372, 2020.

[37] Hui Yin, Yuanhao Gong, and Guoping Qiu. Side window filtering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8758–8766, 2019.

[38] Wang Yin, Peng Lu, Zhaoran Zhao, and Xujun Peng. Yes," attention is all you need", for exemplar based colorization. In *Proceedings of the 29th ACM International Conference on Multimedia*, pages 2243–2251, 2021.

[39] Seungjoo Yoo, Hyojin Bahng, Sunghyo Chung, Junsoo Lee, Jaehyuk Chang, and Jaegul Choo. Coloring with limited data: Few-shot colorization via memory augmented networks. In *Proc. of the IEEE conference on computer vision and pattern recognition (CVPR)*, 2019.

[40] Bo Zhang, Mingming He, Jing Liao, Pedro V Sander, Lu Yuan, Amine Bermak, and Dong Chen. Deep exemplar-based video colorization. In *Proc. of the IEEE conference on computer vision and pattern recognition (CVPR)*, 2019.

[41] Richard Zhang, Phillip Isola, and Alexei A Efros. Colorful image colorization. In *European conference on computer vision*, pages 649–666. Springer, 2016.

[42] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proc. of the IEEE conference on computer vision and pattern recognition (CVPR)*, 2018.

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

[44] Jiaojiao Zhao, Jungong Han, Ling Shao, and Cees Snoek. Pixelated semantic colorization. *International Journal of Computer Vision*, 128, 04 2020.