Iterative Applications of Image Completion with CNN-based Failure Detection

Takahiro Tanaka, Norihiko Kawai*, Yuta Nakashima, Tomokazu Sato, Naokazu Yokoya

8916-5 Takayama, Ikoma, Nara, Japan

Graduate School of Information Science, Nara Instiutute of Science and Technology

Abstract

Image completion is a technique to fill missing regions in a damaged or redacted image. A patch-based approach is one of major approaches, which solves an optimization problem that involves pixel values in missing regions and similar image patch search. One major problem of this approach is that it sometimes duplicates implausible texture in the image or overly smooths down a missing region when the algorithm cannot find better patches. As a practical remedy, the user may provide an interaction to identify such regions and re-apply image completion iteratively until she/he gets a desirable result. In this work, inspired by this idea, we propose a framework of human-in-the-loop style image completion with automatic failure detection using a deep neural network instead of human interaction. Our neural network takes small patches extracted from multiple feature maps obtained from the completion process as input for the automated interaction process, which is iterated several times. We experimentally show that our neural network outperforms a conventional linear support vector machine. Our subjective evaluation demonstrates that our method drastically improves the visual quality of resulting images compared to non-iterative application.

Keywords: Image completion, Image inpainting, Convolutional neural

*Corresponding author
Email address: norihi-k@is.naist.jp (Norihiko Kawai)

network, Failure detection

1. Introduction

Filling missing regions in an image with plausible texture, which is called image completion or image inpainting, is now widely implemented in image editing software thanks to its applicability to various applications ranging from recovering damaged regions to removing people, who might be privacy sensitive, before uploading the image to the web.

Approaches for image completion can be roughly grouped into two types, i.e., the exemplar-based and diffusion-based ones. Recent research efforts have been dedicated to the former as they usually give texture with higher definition. The exemplar-based approach also can be classified into neural network-based and patch-based ones. Although neural network-based approaches have been recently developed, this paper focuses on the patch-based ones, which have been widely investigated for about a dozen years.

The patch-based approaches use image patches extracted from the same or other images in order to synthesize the texture in missing regions. For doing this, most of them solve an optimization problem that involves the pixel values in missing regions as well as similar image patch search. This optimization problem is highly non-convex due to similar image patch search; therefore, it easily ends up with a bad local minimum as shown in the second column from the left in Figure 1. This problem is inevitable, especially if the missing regions are large. That is, the pixels apart from the boundary of a large missing region are almost not constrained by the boundary condition, and thus the algorithm can, e.g., duplicate implausible texture from other regions. One promising remedy is to adopt a human-in-the-loop system, which repeatedly asks the user to label such failure regions until the algorithm gives a satisfactory image. Such a system, however, poses an extra burden to the user. To mitigate the burden, a system is desired that mimics the labeling part by automatically finding failure regions.

Automatic failure detection has been proposed for a specific scenario in [2].

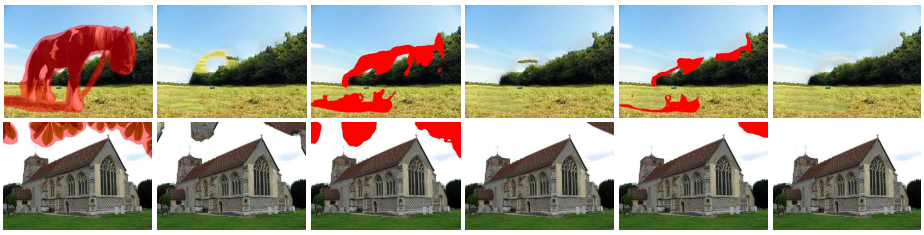

Figure 1: Examples of iterative applications of image completion. Missing regions represented by translucent red in the left-most images are manually labelled. Missing regions represented by solid red in the third and fifth columns from left are automatically labelled by our CNN-based system. Images in the top row are from "Horse at Dyes Farm" and those in the bottom row are "The Church of St. Mary, Lawford, Essex, England (from the southeast)", both of which are licensed under CC BY-SA 2.0 [1].

In panoramic images generated by stitching several images, their peripheral regions are usually not covered by the input images. Thus an image completion algorithm may be applied to such regions to fill them in. However, it may not work well for such peripheral regions, and Kopf et al. [2] automatically crop such failure regions. One important aspect of this work is that failure regions to be cropped are detected by automatic per-pixel quality prediction with a machine learning technique rather than manual labeling. This type of automation is even more advantageous for the human-in-the-loop image completion system as it may require multiple interactions as shown in Figure 1.

This paper presents human-in-the-loop image completion with automatic failure detection. This is a novel framework for boosting the resulting quality of existing image completion algorithms as shown in Figure 1. To eliminate human interaction in a human-in-the loop system, we use a convolutional neural network (CNN) for automatically detecting failure regions. To the best of our knowledge, no existing work addressed this type of automated system for image completion. Our framework can be a breakthrough for enhancing the capability of image completion algorithms by (i) enlarging image regions available for patch-based image completion and (ii) gradually reducing the missing region, which can provide more boundary conditions. To show the advantages

of our system, we experimentally compare it with baseline approaches, including actual human annotators.

The main contributions of this work are summarized as follows.

- A framework for repeatedly applying an image completion algorithm.
- A deep neural network-based approach for finding failure regions in image completion results.
- A set of features tailored for failure region detection in patch-based image completion.
- Experimental comparison to demonstrate that our system is comparable to human-in-the-loop image completion with real human annotators.

2. Related Work

2.1. Image completion

Image completion has been widely studied for past decades, and a number of approaches have been proposed so far, which can be grouped into the exemplar-based and diffusion-based. The diffusion-based approaches [3, 4, 5, 6] minimize an objective involving the derivatives of pixels in missing regions with a certain boundary condition to ensure smoothness on the boundaries of missing regions. They are suitable for filling small regions but are not capable of handling larger ones because they do not synthesize texture in the regions, causing blurry artifacts.

The exemplar-based approaches can be classified into neural network-based and patch-based ones. Neural network-based approaches estimate plausible texture in missing regions using the convolutional neural networks (CNNs) that are trained with pairs of images with and without missing regions. Although neural network-based approaches can deal with only small missing regions initially [7, 8], more detailed textures have become available for large missing regions [9, 10] thanks to generative adversarial networks (GAN) [11].

The patch-based approaches have been investigated for about a dozen years and they synthesize texture in missing regions based on the texture in the rest of the image or other images. Such approaches were originated from successive texture synthesis presented in [12]. This approach copies the pixel values in image patches that are extracted from the rest of the image and are similar to pixels around the missing region's boundary. Since its synthesis results largely depend on the order of copies, a number of later approaches calculate the priority for each pixel on the boundary of the missing region based on structure and sparsity of its texture [13, 14].

Since such approaches still suffer from discontinuities in synthesized regions, an iterative approach was proposed in [15]. This approach casts image completion to an energy minimization problem involving all pixel values in the missing region and similar image patch search. Due to non-convexity of the energy function, the approach iterates the processes to look for the best image patches and to update the pixel values based on the image patches until convergence. In order to avoid bad local minima, the minimization is done in a coarse-to-fine manner, in which an image pyramid is generated and the minimization process is repeatedly applied to the images from the coarsest to the finest. This approach is computationally expensive since it needs to find the best image patches in every single iteration but is drastically reduced by approximating it using PatchMatch algorithm [16].

A number of extensions have been proposed for this iterative approach, mainly for augmenting available texture patterns. The approach in [17] adjusts the brightness of pixels before copying them to the missing region to use the same texture pattern in different lighting conditions. The approaches in [18, 19, 20] handle geometric variations of texture by flipping, rotating, and scaling image patches. Further extension has been conducted in [21], which synthesizes textures considering perspective distortions on the assumption that many scenes consist of multiple planes and each plane has parallel lines. These extensions basically give a better image completion results; however, the additional parameters for handling geometric transformations cause extra local

minima. For example, rotated roof patterns are copied to the sky region as shown in Figure 1.

2.2. Evaluation of image completion quality

Image quality evaluation without ground truth images has been studied in the field of image synthesis, including image completion, for objectively picking out the best image completion result out of those by several image completion approaches or for finding visual artifacts to be compensated later.

Dang et al. [22] designed an evaluation metric for image completion based on the visual coherence and the visual saliency. Voronin et al. [23] evaluated image completion results using an SVM regressor trained with pairs of image completion results and manually annotated scores. Since these methods give a single evaluation result for an image, they are inappropriate for our approach, which requires pixel-wise quality evaluation.

To the best of our knowledge, the approach by Kopf et al. [2] is the only one that evaluates image completion results in a pixel-wise manner. Their method is mainly designed to crop a panoramic image generated by image stitching together with image completion for peripheral regions that are not covered by the captured images. To retain as large portion of the stitched image as possible but with less visual artifacts, they propose to evaluate per-pixel image completion quality using a variant of AdaBoost classifier with low-level features, such as color, edge density, etc. For general image synthesis, an approach with pixel-wise evaluation has been proposed using low-level features [24].

Ours can also be viewed as an approach to evaluate the pixel-wise image completion quality as it classifies each pixel into success/failure. Being different from Kopt et al.'s, which evaluates the quality only once, our pixel-wise classification is embedded into the human-in-the-loop system in order to mimic the human interaction in it. In addition, we use a set of feature maps, such as pattern similarities, that are tailored for our image completion system.

Figure 2: Example implausible textures obtained by patch-based methods. From left to right, input images, results by the method in [20], results by the method in [19]. Original images are from "Летний отдых", "Horse at Dyes Farm", "good mood", and "Wolverton Manor Garden Fair 2013 - 05" licensed under CC BY 2.0 [25] or CC BY-SA 2.0 [1].

3. Overview

The patch-based image completion algorithms usually synthesize plausible textures in the scene. However, they can duplicate implausible texture especially for large missing regions, in which the boundary condition is often insufficient to suppress appearance of implausible texture, as shown in Figure 2.

A human-in-the-loop image completion system can handle such failures with a help of the user, although such a system has not been published nor, at least, spotlighted in the community so far in spite of its advantage in resulting images' quality. Our approach replaces the human part of such a system with a CNN: Given an image with missing regions Ω, we first do image completion. Our CNN then classifies each pixel in Ω into success or failure. To pixels that are classified as failure, we again apply the image completion algorithm with additional exemplars of pixels that are classified as success. The classification and image completion processes are repeated for M times according to our iteration strategy.

Image completion has been widely studied so far, and there are a number of algorithms. Our approach is not dependent on a specific image completion algorithm as long as it is patch-based, so any algorithm can be used. In this work, we employ our previous method in [19] as one of the state-of-the-art methods. Although this method usually gives results with good quality as shown in the paper [19], it sometimes gives implausible textures as shown in Figure 2,

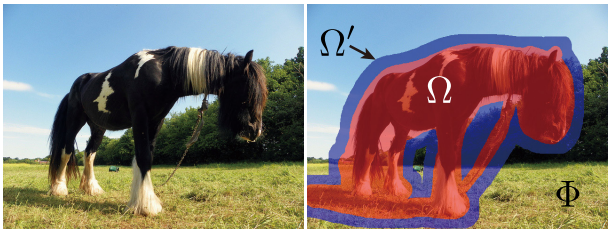

Figure 3: Definitions of missing region Ω, expanded missing region Ω', and data region Φ.

as other state-of-the-art patch-based completion methods do. We briefly review the algorithm in the next section to make the paper self-contained.

4. Image Completion Algorithm

Let Ω' be the region that contains pixels covered by a kernel K centered at any $i \in \Omega$ as shown in Figure 3. Region Φ is the complement of Ω', referred to as the data region. Most patch-based image completion algorithms are formulated as an optimization problem, which minimizes an energy. The method [19] uses the following energy, which encodes that (i) the pixel values in Ω are copied from Φ and (ii) similar patterns usually lie in nearby regions:

$$E = \sum_{i \in \Omega'} \min_{j, H_{ij}} w_i [\mathrm{SSD}(i, j, H_{ij}) + \kappa E_{\mathrm{R}}(i, j)], \tag{1}$$

where w_i is the weight for pixel i, which is based on the distance from the boundary and the texture complexity; κ is a predetermined parameter to control the contribution of each term.

$\mathrm{SSD}(i, j, H_{ij})$ is based on the sum of squared differences of pixels around i and j, after certain local geometric transformation H_{ij} around pixel j. In their method, H_{ij} is identity, horizontal flip, or vertical flip. Letting $\mathbf{x}_i$, $\mathbf{x}_j \in \mathbb{R}^2$ be the positions of pixels i and j, and $I(\mathbf{x}_i)$ the pixel value at $\mathbf{x}_i$, SSD is defined as

$$\mathrm{SSD}(i, j, H_{ij}) = \sum_{p \in K} \| I(\mathbf{x}_i + \mathbf{p}) - \alpha_{ij} I(\mathbf{x}_j + H_{ij}(\mathbf{p})) \|^2, \tag{2}$$

where α_{ij} is for adjusting the brightness of the patch.

E_{R} is a regularization term to encourage copy from nearby pixels, which is given using the sigmoid function by

$$E_{\mathrm{R}}(i,j) = \begin{cases} \frac{|K|}{1+e^{-a\|\mathbf{x}_i - \mathbf{x}_j\|+b}} & \text{if } H_{ij} \text{ is identity} \\ G & \text{otherwise} \end{cases}, \tag{3}$$

where $|K|$ is the number of pixels in K, a and b are predetermined constant, and G is a large constant. Please refer to [19] for more detail.

We minimize the energy in Equation (1) by iterating two processes. The first process is to solve the inner minimization problem, which finding an image patch centered at j that is the most similar to the patch centered at $i \in \Omega'$. We can approximately solve this problem by the PatchMatch algorithm [26], which drastically reduces the computational cost. The second process is to calculate a weighted mean of pixel values in the patches that are the most similar to patches that include i for pixel value $I(\mathbf{x}_i)$ for each $i \in \Omega$.

5. Finding failure pixels using CNN

The above minimization problem is highly non-convex, which may get stuck into a bad local minimum. To avoid this, users of image completion algorithms may begin additional rounds of image completion with manual interaction to identify failure regions. In order to automate this interaction, we design and train a CNN that classifies each pixel in the original missing region Ω into success or failure.

5.1. Data

When users of the human-in-the-loop image completion system label failure pixels, they only see the resulting image. This is because users can exploit the semantics of the image. However, finding failure pixels is tough for machines due to the incapability of considering the semantics from just the resulting image. In order to make up for the missing semantics, besides the image after the initial image completion, we consider additional data that can be obtained from the image completion process, i.e., RGB image after image completion, SSD map,

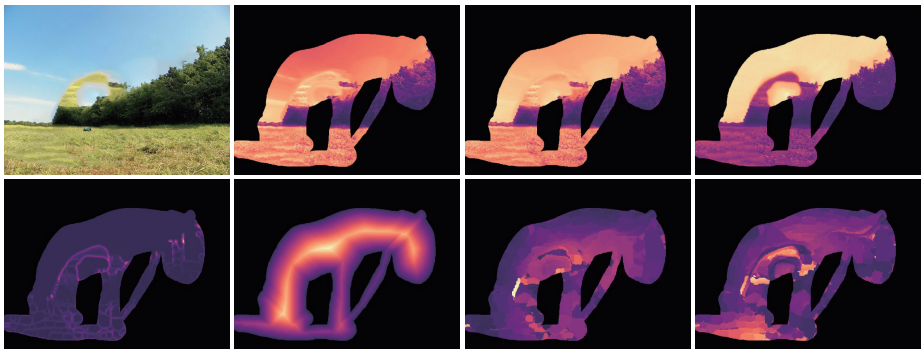

Figure 4: An example of feature maps (brighter colors are higher values). From top-left to bottom-right: Resulting image and its red, blue, and green channels; SSD map, distance-to-boundary (DB) map, and distance-to-similar-patch (DSP) maps for the major and minor axes.

distance-to-boundary (DB) map, and distance-to-similar-patch (DSP) maps as shown in Figure 4. From which, we select features used for finding failure pixels by exhaustively evaluating all the combinations.

SSD map stores the dissimilarity SSD between the image patches around pixel $i \in \Omega$ and pixel $j \in \Phi$, which is used in most image completion algorithms. A failure region usually contains pixels with a larger SSD value as shown in Figure 4, because the minimization cannot find good image patches that smoothly connect different parts in that region. This indicates that SSD can be a good cue for finding failure regions. Our case uses the modified SSD with the brightness adjustment coefficient defined in Equation (2).

DB map stores the minimum distance between a pixel in a missing region and a pixel in data regions. Generally, image completion failure is more likely to occur for pixels far from the boundary between missing region Ω and data region Φ. This is because these pixels are hardly constrained by the surrounding data region and thus suffer from the ambiguity in finding similar patches. For such pixels, the image completion algorithms may put implausible objects (e.g., building roofs in the sky in Figure 1). In order to find such pixels, we use a DB map.

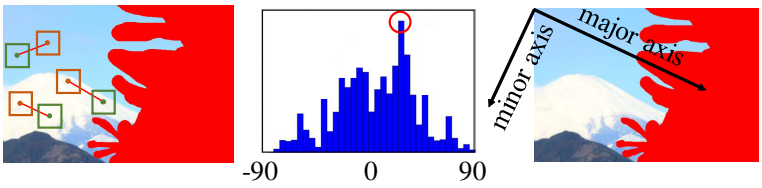

Figure 5: Illustration of fiding axes for DSP maps. From left to right: Pairs of similar patches, histogram of directions, and determined major and minor axes.

DSP maps stores the distances between each pixel and its associated image patch along two axes. The idea for the maps comes from the fact that an object usually has consistent texture on it and thus texture in images is locally consistent. Due to this texture locality, the image completion algorithms usually pick out image patches to be copied to pixels from their closer regions; otherwise, resulting images may have locally inconsistent texture. Therefore, We use distances between each pixel and its associated image patch as this texture locality to facilitate our classification task.

The relative direction from each pixel to its associated image is also informative although the absolute directions (i.e., directions with respect to the image's horizontal and vertical axes) are not always informative for this task because people may take photos with a camera rotated. Therefore, we find the major and minor axes, where the major axis is the direction that the most of similar patches are likely to lie in and the minor axis is its orthogonal. For doing this, our approach calculates the histogram of the directions from pairs of the most similar patches for all pixels in data region Φ and uses the the most voted direction as the major axis as shown in Figure 5. To speed up the histogram calculation, we resize the input image so that the length of its shorter edge is 200 pixels.

5.2. Network architecture

For success/failure labeling for pixel i in missing region Ω, we extract the 56×56 image region centered at pixel i, which is referred to as sub-region R_i.

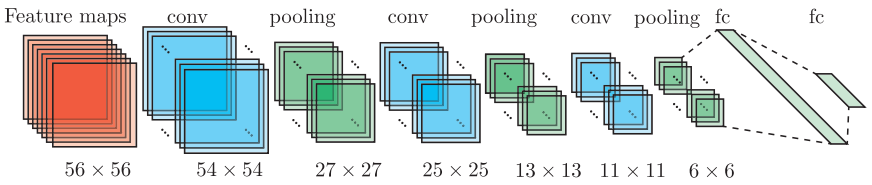

Figure 6: Network architecture. "conv," "pooling," and "fc" stand for convolution, max pooling, and fully connected layers, respectively. The size of each layer is also shown below it.

The sub-region R_i has 7 channels consisting of RGB maps (3 channels), SSD map (1 channel), DB map (1 channel), and DSP maps (2 channels).

Thanks to this rich input data, we use a relatively shallow network architecture shown in Figure 6. Our network has three 3×3-kernel convolution layers, each of which has 30 channels and is followed by a 2×2 max pooling layer and ReLU non-linearlity [27]. On top of this, it has two fully connected layers: the lower layer has 100 units with ReLU non-linearily, and the upper layer has 2 units with softmax outputs corresponding to success and failure, respectively.

5.3. Training

For training our network, we collected 1,078 images from the web. Certain objects (mostly a foreground object) in these images were marked as missing regions by the authors and the image completion algorithm [19] was applied. Human annotators (the authors and some students) then manually labeled success/failure pixels subjectively. The data was augmented by horizontal flip and resizing. We then extracted 146,556 pairs of 56×56 sub-regions and success/failure labels from them. For validation, we collected other 78 images and extracted 11,000 pairs in the same manner. The ratios of extracted success/failure examples in the training and validation sets are 50% and 50%.

We used softmax cross entropy as the loss for training. To alleviate overfitting, the dropout technique [28] was applied prior to the fully connected layers. We adopted the stochastic gradient descent (SGD) algorithm for training the network parameters, where the mini-batch size was 150. We stored the network

Figure 7: An example of a problem caused by labeling failure pixels with $\theta = 0.5$. From left to right: Example image with a missing region, initial image completion result, an automatically labeled failure region ($\theta = 0.5$), image completion result of the left image. Note that there is undesirable texture. The original is from "Picknick kann los gehen!" licensed under CC BY 2.0 [25].

parameters that gave the best classification performance over the validation data set.

6. Iteration Strategy

In the human-in-the-loop system, the image completion process and the success/failure labeling process are iterated in turn for several times until the user gets a satisfying result. Our automated system also follows this flow but until it gets no failure labels or reaches the predetermined number M of iterations.

We use softmax outputs with two possible labels (success/failure). As demonstrated in Figure 7, if there are pixels that are labeled with success in failure regions (as in the image at the third column in Figure 7), the image completion process is constrained by them and results in implausible texture because pixels labeled with success are fixed and used as exemplars in subsequent iterations. To prevent this problem, we apply the following heuristic thresholds θ to the output corresponding to the success label for iteration $m = 1, \ldots, M$ after the initial completion.

$$\theta_m = \begin{cases} 0.5 & \text{for } M = 1 \\ \frac{0.5-T}{M-1}(m-1) + T & \text{for } M > 1 \end{cases}, \tag{4}$$

where T is a predetermined parameter. The threshold θ_m gradually increases as the iteration goes on, which is more conservative than just using $\theta = 0.5$ since

extra application of image completion to success pixels usually does not cause quality degradation (although each iteration needs to reduce the failure regions because the result does not change otherwise).

7. Experimental Results and Discussion

We first show the classification performance, which means how well our automated system mimics the human annotators who label the image completion results. Since the classification performance does not necessarily correlate to the perceptual quality of resulting images, we show the results of our user study to evaluate the quality. For implementing the CNN, we used the Caffe framework [29]. In the experiments, we used 25 test images in which target objects were manually labeled by the authors that are different from the images for training and validation.

7.1. Classification performance over validation set

We evaluated the classification performance of our CNN-based classifier using the accuracy rate over our validation set in order to spot good combinations of features. For comparison, we also evaluated SVM-based classifiers that take a flattened and concatenated sub-region as features. For the CNN-based classifiers, we trained them four times and took the best one because the training process is stochastic due to the SGD algorithm, although the accuracy rates were relatively stable over the four runs of training. Note that we limit the number of iterations in the SVM training to 1,000 for the case when SVM training does not converge for a long time. We used LIBLINEAR [30] with the penalty parameter $C = 1$.

Table 1 shows the results. Combination 1 that uses all maps clearly outperformed the other combinations. Among combinations that use a single type of maps (i.e., combinations 8, 12, 14, and 15), combination 14's accuracy is lower than the others. This implies that the DB map is less useful by itself for this classification task. Also, the low accuracy rate for combination 8, which uses

Table 1: Classification accuracy in percentage. "✓" stands for that corresponding maps are used in that combination.

Comb. #	RGB	SSD	DB	DSP	CNN	SVM
1	✓	✓	✓	✓	**84.02**	70.34
2	✓	✓	✓		81.62	64.14
3	✓	✓		✓	81.18	73.88
4	✓	✓			76.94	64.05
5	✓		✓	✓	81.72	59.67
6	✓		✓		78.50	56.47
7	✓			✓	79.09	**74.28**
8	✓				74.46	54.52
9		✓	✓	✓	81.81	72.60
10		✓	✓		79.84	68.14
11		✓		✓	81.49	59.73
12		✓			79.04	68.65
13			✓	✓	79.09	59.63
14			✓		63.57	59.81
15				✓	78.58	60.80

only RGB maps (i.e., image completion results), demonstrates the advantages of our feature maps tailored for this classification task. Our CNN-based classifiers mostly outperformed the SVM-based classifiers, which clearly demonstrates its advantage, at least for the validation set.

We then evaluated the best three combinations (combinations 1, 5, and 9) using the 25 test images by the receiver operating characteristic (ROC) curve and the area under the ROC curve. Figure 8 shows the results. Interestingly, combination 9 outperformed the others in terms of AUCs without using RGB maps. We consider this is because we used the test image set for the evaluation in which the ratio of success/failure for the entire missing regions is 70% and 30%. This is different from the ratios for training and validation sets. From the result, we use the CNN parameters trained with combination 9 in the following

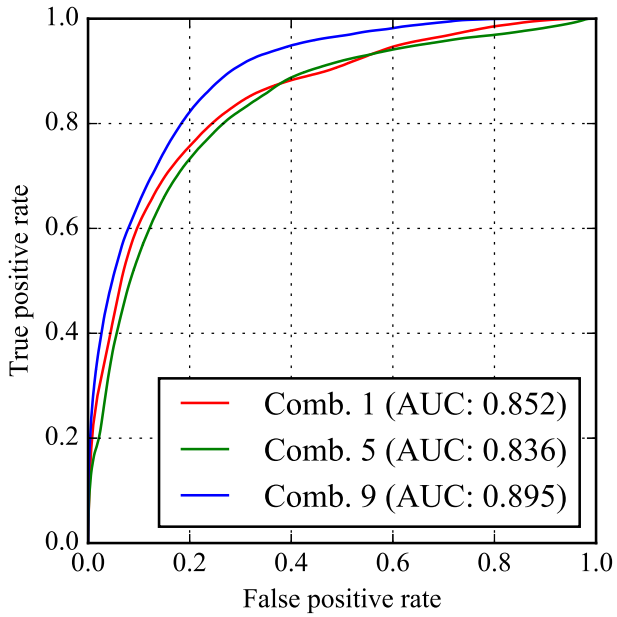

Figure 8: ROC curves and AUCs for combinations 1, 5, and 9.

section.

7.2. Parameters for iteration strategy

To show the effects of the parameters for iteration strategy (i.e., threshold parameter T and number M of iterations), we ran our automated system for our 25 test images with different parameter values.

We first show how T affects the results. We applied our system to the test images with T ranging from 0.2 to 0.5 by step size 0.05 and M being fixed to 3. Figure 9 shows an example used for this experiment and Figure 10 shows example results with different T for the image in Figure 9. This demonstrates that $T = 0.2$ is too conservative and the failure regions hardly reduce. Since the missing region and the failure region were similar, the results of initial and first-round image completion were also similar. On the other hand, $T = 0.5$ looks too optimistic and some failure pixels were not marked as failure. This is worse than being conservative because our current system never changes pixels once they are marked as success and in turn they constrain image completion

Figure 9: An example of test images with missing regions (left) and their initial image completion results (right) used for examining how T affects. The original image is from "Harbor Fish Market" licensed under CC BY 2.0 [25].

Figure 10: Examples of failure pixel labeling and image completion with different T. From left to right: $T = 0.2$, 0.35, and 0.5 for the image in Figure 9. From top to bottom: first-round failure pixel labeling results, first-round image completion results, and third (final)-round image completion results.

of regions they touch. The iteration strategy with $T = 0.35$ worked better than others. Figure 11(left) shows the relationship between T and the mean of the

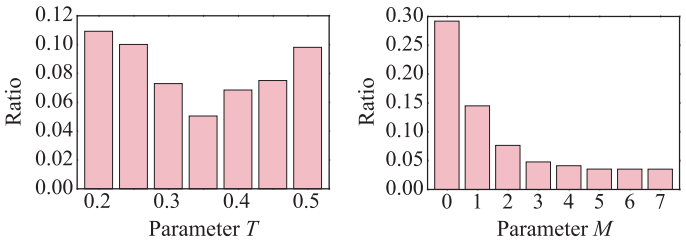

Figure 11: Parameter T (left) and parameter M versus the ratio of the area of undesirably textured regions to the area of the original missing regions.

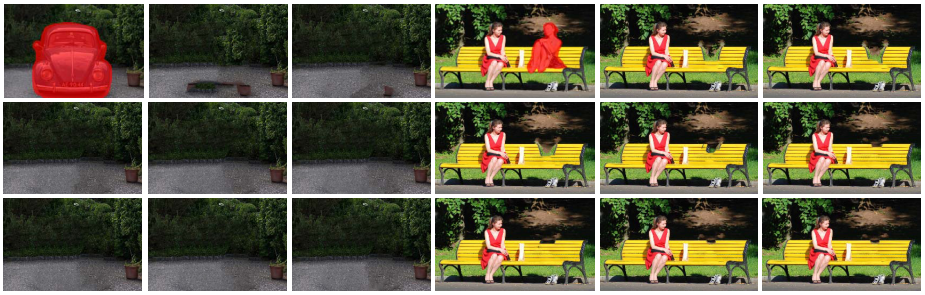

Figure 12: From left to right, top to bottom: Original images with missing regions, initial image completion results, results after final-round image completion for $M = 1$, 2, 3, 4, 5, 6, and 7. The left images are from "Жук" and the right image are from "Летний отдых", both of which are by akras, licensed under CC BY 2.0 [25].

ratios of the area of undesirably textured regions in the final completion result to the area of original missing regions for the 25 test images (undesirably textured regions were manually labeled by human annotators). As expected, $T = 0.35$ got smallest undesirably textured regions.

We also applied our system to the test images with different M ranging from 1 to 7 with $T = 0.35$. Figure 12 shows examples. The relationship between M and the mean of the ratios of the area of undesirably textured regions to the area of the original missing regions for the 25 test images are shown in Figure 11(right). As the result imply, the ratio almost converges around $M = 5$.

Our automated system thus iterates the failure pixel labeling and image completion processes five times with $T = 0.35$ after the initial completion. Figure 13 shows an example run of our automated system. Results for all our 25

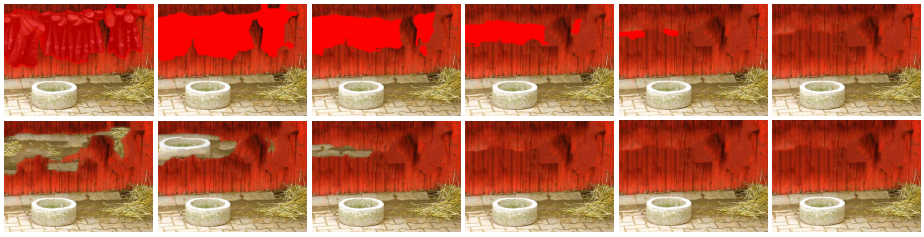

Figure 13: Example run of our automated system with step-by-step progress. The left-most image in the top row is original image with a missing region. Other images in the top row are success/failure labeling results of their lower-left images. The bottom row contains image completion results for their above images. The original is from "Line of Boots" licensed under CC BY 2.0 [25].

test images, together with the results by the state-of-the-art method [20], can be found in the supplementary material, from which we can know that the proposed iterative system generated more plausible textures compared with results obtained by applying the image completion methods in [19] and [20] only once.

7.3. User study

To evaluate the quality of our automated system's results in an objective way, we conducted user study, in which we compared the following five approaches that use the method [19] as a patch-based image completion method to focus on the effectiveness of the iterative system:

(i) Image completion by the method [19]

(ii) Iterative applications of the method [19] with uniformly reducing the missing region from outer to inner so that the missing region disappears after five iterations of labeling and completion after the initial completion as shown in Figure 14

(iii) Our system ($T = 0.5$ and $M = 1$)

(iv) Our system ($T = 0.35$ and $M = 5$)

(v) Iterative applications of the method [19] with manual labeling by a human annotator (up to five iterations of labeling and completion after the initial completion for fair comparison)

Figure 15 shows some examples of images after image completion by above approaches that were used in our user study (the supplementary material contains all images). In this user study, we randomly presented the test image processed by a certain approach and asked 18 subjects (13 males and 3 females in their twenties who are students as well as 1 male and 1 female in their fifties) if each image can be used for books and magazines as a material, and they rated the images with scores from 1 (completely refuse to use) to 5 (be pleased to use).

Figure 16 shows the box plots of the ratios of undesirably textured regions in the final completion results to the area of original missing regions and the scores for the five approaches. Our automated system (iv) largely outperforms approaches (i) to (iii), and is comparable to iterative application with manual labeling (v), which can be deemed as an upper bound for this image completion algorithm.

7.4. Discussion

From the result of our qualitative evaluation (Figure 16, left) and user study (right), our automated system (iv) clearly outperformed the others except (v). Comparing (ii) and (iv), we confirmed that the quality of image completion results is quit different depending on the labeling even if the number of iterations is the same. Comparing (iv) with (v), we found that (v) is advantageous when an image contains few regions in data regions whose texture is similar to the texture of missing regions. This is because our system sometimes labels success pixels as failure (due to $T = 0.35$), while the human annotators labels as small regions as possible. Consequently, the algorithm with human annotations can use larger data regions to fill smaller regions. Especially, when applying image completion to the coarsest level in the coarse-to-fine approach, larger missing regions hinder the algorithm from finding the best patch because it may be

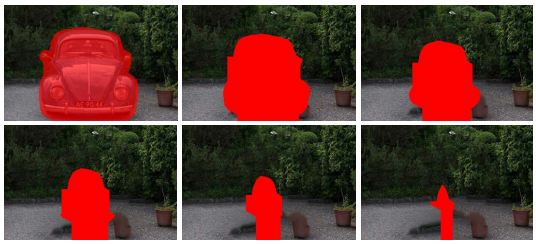

Figure 14: Illustration of labeling a missing region in each iteration by approach (ii). From top-left to bottom-right: Initial missing region, missing regions in the first to fifth iterations.

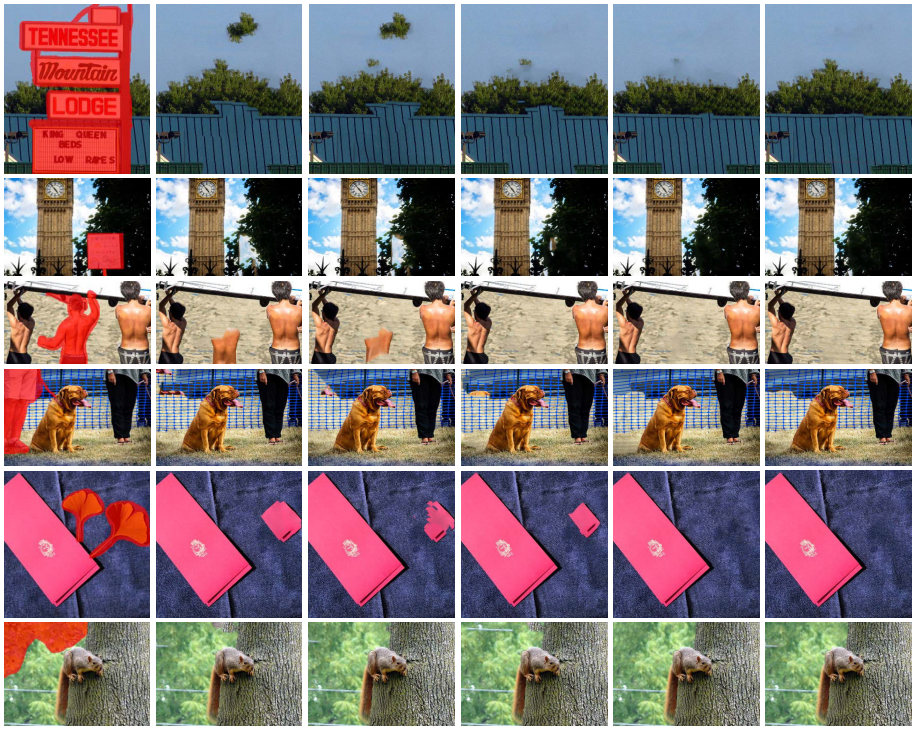

Figure 15: Examples of the images used in our user study. From left to right: Original image with a missing region, image completion results by approaches (i) to (v). The original images are from "Tennessee Mountain Lodge sign", "March for Refugees September 2015 - 12", "Boys of Summer / Lingnan University Rowing Team / Hong Kong Water Sports / SML.20130809.7D.49261", "Wolverton Manor Garden Fair 2013 - 05", "Picknick kann los gehen!", and "Squirrel 03" licensed under CC BY 2.0 [25] or CC BY-SA 2.0 [1].

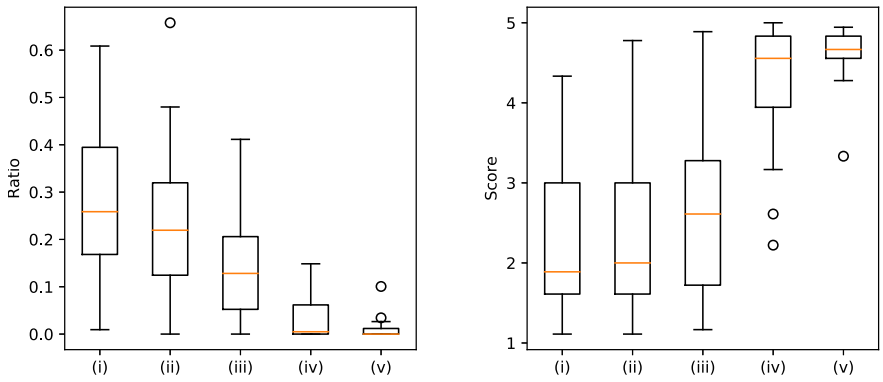

Figure 16: Left: The ratio of undesirably textured regions in the final completion result to the area of original missing regions for each approach. Right: User study results. Both plots show 25% (bottom edges of boxes), 50% (red bars), and 75% (top edges of boxes), as well as the maximum and minimum values (whiskers) excluding outliers (circles)

included in Ω' which is the expansion of missing region Ω. The inappropriate correspondences in the coarsest level are often kept in the finest level, resulting in implausible texture.

One major limitation of our automated system is that it does not consider the content of an image. For example, an image completion algorithm can duplicate the moon if the image contains moon and the missing region is sufficiently large. For such a case, human annotators can label the duplicated moon as failure, but our system may not because there might be no evidence of failure in our feature maps. This is a very challenging problem as it requires knowledge of the scene or common sense. Recent research efforts have been dedicated to such problems [31], so our next step can be to integrate such approaches to our system to boost the performance.

8. Conclusion

A human-in-the-loop image completion system is a promising approach to get higher quality image completion results at the cost of extra burden of the

user. To unload the user's burden, we proposed a framework of human-in-the-loop style image completion with automatic detection of failure regions using CNN trained with a large amount of data tailored for patch-based image completion. Our experimental results demonstrated that our automated system is comparable to the human-in-the-loop system with real human annotators. Our future work includes to seek for a better network architecture, such as a fully convolutional network [32] and deconvolution [33, 34], possibly with integrating an approach for semantic acquisition, as well as increase the number of training data for a deeper network architecture.

Acknowledgements

This work was partially supported by Grants-in-Aid for Scientific Research Nos. 15K16039, 16H06302, 18H03273, and 18K11375 from the Japan Society for the Promotion of Science (JSPS).

References

[1] Creative Commons, Attribution-sharealike 2.0 generic (cc by-sa 2.0), `https://creativecommons.org/licenses/by-sa/2.0/deed.en`.

[2] J. Kopf, W. Kienzle, S. Drucker, S. B. Kang, Quality prediction for image completion, ACM Transactions on Graphics 31 (6) (2012) 1.

[3] M. Bertalmio, G. Sapiro, V. Caselles, C. Ballester, Image inpainting, in: Proc. SIGGRAPH 2000, 2000, pp. 417–424.

[4] T. F. Chan, J. Shen, Non-texture inpainting by curvature driven diffusion (CDD), Visual Communication and Image Representation 12 (4) (2001) 436–449.

[5] C. Ballester, M. Bertalmio, V. Caselles, G. Sapiro, J. Verdera, Filling-in by joint interpolation of vector fields and gray levels, IEEE Transactions on Image Processing 10 (8) (2001) 1200–1211.

[6] J. Shen, S. H. Kang, T. F. Chan, Euler's elastica and curvature-Based inpainting, SIAM Journal on Applied Mathematics 63 (2) (2003) 564–592.

[7] J. Xie, L. Xu, E. Chen, Image denoising and inpainting with deep neural networks, in: Proc. International Conference on Neural Information Processing Systems, Vol. 1, 2012, pp. 341–349.

[8] D. Eigen, D. Krishnan, R. Fergus, Restoring an image taken through a window covered with dirt or rain, in: Proc. International Conference on Computer Vision, 2013, pp. 633–640.

[9] D. Pathak, P. Krähenbühl, J. Donahue, T. Darrell, A. Efros, Context encoders: Feature learning by inpainting, in: Proc. IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 2536–2544.

[10] S. Iizuka, E. Simo-Serra, H. Ishikawa, Globally and Locally Consistent Image Completion, ACM Transactions on Graphics 36 (4) (2017) 107:1–107:14.

[11] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. C. Courville, Y. Bengio, Generative Adversarial Nets, in: Proc. International Conference on Neural Information Processing Systems, 2014, pp. 2672–2680.

[12] A. A. Efros, T. K. Leung, Texture synthesis by non-parametric sampling, in: Proc. IEEE International Conference on Computer Vision, Vol. 2, 1999, pp. 1033–1038.

[13] A. Criminisi, P. Pérez, K. Toyama, Region filling and object removal by exemplar-based image inpainting, IEEE Transactions on Image Processing 13 (9) (2004) 1200–1212.

[14] Z. Xu, J. Sun, Image inpainting by patch propagation using patch sparsity, IEEE Transactions on Image Processing 19 (5) (2010) 1153–1165.

[15] Y. Wexler, E. Shechtman, M. Irani, Space-time completion of video, IEEE Transactions on Pattern Analysis and Machine Intelligence 29 (3) (2007) 463–476.

[16] C. Barnes, E. Shechtman, A. Finkelstein, D. B. Goldman, PatchMatch: A randomized correspondence algorithm for structural image editing, ACM Transactions on Graphics 28 (3) (2009) 24:1–24:11.

[17] N. Kawai, T. Sato, N. Yokoya, Image inpainting considering brightness change and spatial locality of textures and its evaluation, in: Proc. Pacific Rim Symposium on Image and Video Technology, 2009, pp. 271–282.

[18] A. Mansfield, M. Prasad, C. Rother, T. Sharp, P. Kohli, L. V. Gool, Transforming image completion, in: Proc. British Machine Vision Conference, 2011, pp. 121.1–121.11.

[19] N. Kawai, N. Yokoya, Image inpainting considering symmetric patterns, in: Proc. International Conference on Pattern Recognition, 2012, pp. 2744–2747.

[20] S. Darabi, E. Shechtman, C. Barnes, D. B. Goldman, P. Sen, Image melding: Combining inconsistent images using patch-based synthesis, ACM Transactions on Graphics 31 (4) (2012) 82:1–82:10.

[21] J.-B. Huang, S. B. Kang, N. Ahuja, J. Kopf, Image completion using planar structure guidance, ACM Transactions on Graphics 33 (4) (2014) 129:1–129:10.

[22] T. T. Dang, A. Beghdadi, C. Larabi, Perceptual quality assessment for color image inpainting, in: Proc. IEEE International Conference on Image Processing, 2013, pp. 398–402.

[23] V. Voronin, V. Marchuk, E. Semenishchev, S. Maslennikov, I. Svirin, Inpainted image quality assessment based on machine learning, in: Proc. International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision, 2015, pp. 167–172.

[24] R. Herzog, M. Čadík, T. O. Aydın, K. I. Kim, K. Myszkowski, H.-P. Seidel, NoRM: No-reference image quality metric for realistic image synthesis, Computer Graphics Forum 31 (2) (2012) 545–554.

[25] Creative Commons, Attribution 2.0 generic (cc by 2.0), `https://creativecommons.org/licenses/by/2.0/deed.en`.

[26] C. Barnes, E. Shechtman, D. B. Goldman, A. Finkelstein, The generalized patchmatch correspondence algorithm, in: Proc. European Conference on Computer Vision, 2010, pp. 29–43.

[27] V. Nair, G. E. Hinton, Retified linear units improve restricted Boltzmann machines, in: Proc. International Conference on Machine Learning, 2010, pp. 807–814.

[28] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov, Dropout: A simple way to prevent neural networks from overfitting, Journal of Machine Learning Research 15 (2014) 1929–1958.

[29] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, T. Darrell, Caffe: Convolutional architecture for fast feature embedding, in: Proc. ACM International Conference on Multimedia, 2014, pp. 675–678.

[30] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, C.-J. Lin, LIBLINEAR: A library for large linear classification, Journal of Machine Learning Research 9 (2008) 1871–1874.

[31] R. Vedantam, T. Lin, X.and Batra, C. Lawrence Zitnick, D. Parikh, Learning common sense through visual abstraction, in: Proc. International Conference on Computer Vision, 2015, pp. 2542–2550.

[32] E. Shelhamer, J. Long, T. Darrell, Fully convolutional networks for semantic segmentation, IEEE Transactions on Pattern Analysis and Machine Intelligence 39 (4) (2017) 640–651.

[33] O. Ronneberger, P. Fischer, T. Brox, U-net: Convolutional networks for biomedical image segmentation, in: Proc. Medical Image Computing and Computer-Assisted Intervention (MICCAI), Vol. 9351, 2015, pp. 234–241.

[34] H. Noh, S. Hong, B. Han, Learning Deconvolution Network for Semantic Segmentation, in: Proc. International Conference on Computer Vision, 2015, pp. 1520–1528.