*International Journal of Industrial and Manufacturing Systems Engineering*
2025, Vol. 10, No. 1, pp. 9-19
https://doi.org/10.11648/j.ijimse.20251001.12

Research Article

# U-Paint: Image Inpainting and Object Detection via Partial Convolutions

**Rahul Singh*, Martin Bruder, Akshay Kumar, Rohan Joshi**

Language Technology Institute, Carnegie Mellon University, Pittsburgh, Pennsylvania

## Abstract

Image inpainting and object detection are two well-established research areas with significant real-world applications in computer vision, digital forensics, media editing, and augmented reality. Image inpainting is a technique used to restore missing, distorted, or removed sections of an image, often to recreate its original appearance. It can also be applied to remove objects from an image while reconstructing the background in a visually coherent manner. Traditional inpainting methods relied on manual editing or simple interpolation techniques, whereas modern deep learning-based approaches utilize convolutional neural networks (CNNs) and generative adversarial networks (GANs) to achieve realistic results. Object detection, on the other hand, involves identifying and localizing specific objects, such as people, buildings, or vehicles, within digital images and videos. This paper presents a system that integrates object detection with image inpainting to automate object removal and image restoration. By detecting an object and applying advanced inpainting techniques, the system seamlessly reconstructs the image without noticeable artifacts. The proposed approach has broad applications in image editing, surveillance, content moderation, and privacy protection, providing an effective and automated solution for object removal and background restoration.

## Keywords

Image Inpainting, Object Detection, Partial Convolution, U-Net, Mask R-CNN, Semantic Image Segmentation

## 1. Introduction

Inspired by the applications of deep learning in the field of media manipulation, the goal of this research is to build a system that can detect an object and reconstruct the image without that object in it while retaining its surrounding environment. Our main innovation is to combine and attempt to advance two existing technologies - Object Detection and Image Holes Inpainting. There are several possible extensions to the application in question; for example, video inpainting. These applications also intersect with broader questions of human adjudication and AI-assisted decision- making [14].

The primary motivation behind implementing this deep learning system is to augment an understanding of convolutional neural networks to advance the present state-of-the-art in an up- and-coming application of photo-manipulation. Our objective is to develop the model for image inpainting, and to combine this with the object detection technique, as elucidated in this paper. In some sense, our work is a re-implementation of Nvidia's paper on "Image Inpainting with Partial Convolutions" [1], and we extend this by adding object detection to it. An example of the application of NVIDIA's technique to an image is shown in Figure 1. We use a Region-based Convolutional Neural Network (R-CNN)

*Corresponding author: rahuls2@alumni.cmu.edu (Rahul Singh)

**Received:** 18 February 2025; **Accepted:** 6 March 2025; **Published:** 13 September 2025

to mask out identified objects in an image, and the concept of Partial Convolutions applied to a modified U-Net model to manipulate those sections of the image.

As described in the following sections, the mask R-CNN for image-masking has been implemented, and the custom layers and the loss function have been developed for image inpainting. Besides the goal of building a system to detect and reconstruct objects in images, the final product will also incorporate extensions such as the ability to replace an existing object in an image belonging to a particular class with a specified object from a different class. We also plan to work on extending U-Paint to include additional features and address other related applications. These extensions to our work are expounded in the penultimate section of the paper.

The following account details the model in use. A brief discussion about the related literature is followed by an explanation about the experiments conducted and an analysis of the results. The final section provides a conclusion to our work along with a summary of the key learning from this exercise.

(a) Original Image (b) Inpainted Image

***Figure 1.*** *An example of NVIDIA's Image Inpainting application [1]. Subfigure (a) is the original image. After masking out the people in the background, the area covered by the mask in the image is inpainted and the result is seen in subfigure (b).*

## 2. Related Work

Notable progress has been made in the field of image inpainting over the last two decades. Early implementations include diffusion-based and patched-based methods, of which the latter arguably produced the best results. In Sun et al. [2], structure propagation along user specified curves is used for image inpainting. This is followed by a patch-based texture synthesis to fill the remaining portion of the region to be inpainted. If only a single curve is specified, dynamic programming is used for structure propagation, whereas the Belief Propagation algorithm [3] is used when more curves are available. The limitation of the system is that in order for it to capture enough surrounding information to inpaint the desired region, the user needs to specify curves that extend from the region to be inpainted to the region outside. Another significant patch-based system is PatchMatch [4] which uses a fast, randomized algorithm approach to optimally find approximate nearest neighbor matches between image patches. However, these systems are not sensitive to the semantics of an image and are too slow for real-time applications. Related work on learning semantic face attributes underscores the value of semantic priors in visual tasks [13].

Deep Learning approaches have been implemented to deal with these problems. Kohler et al. [5] uses Multi-Layer Perceptrons (MLP) to learn a mapping between corrupted image patches with missing pixels and their corresponding complete image patches. This approach shows that the shape of the missing region or "hole" provides valuable information for the inpainting process. During training the input features depend on the shape of the hole or masks. This idea is extended to Blind inpainting in which the exact shape and size of the hole is not known. However, this system only works well with small holes. In the case of larger holes, it tends to produce blurry images which could be due to the limited number of pixels that the nonlinear filter, learnt by the MLP, can extend.

Yeh et al. [6] was able to achieve substantial results in image inpainting even when the holes are large. Given a Generative Adversarial Network (GAN) trained on real images, the closest representation to the input image with a hole, is found in the latent image manifold by iteratively updating the given image's representation. This is then used by the GAN to fill in the hole. It is evident that this approach relies heavily on the quality of the GAN and how it is trained. Ulyanov et al. [7] shows that the structure of a generative network captures a sufficient level of detail of the images, such that image inpainting can be done without any prior learning.

These approaches use standard convolutions which are conditioned on both the pixels in the known region as well as the values in the masked hole region. This can lead to undesirable features in the inpainted images. To overcome this issue Liu [1] uses partial convolutions that are conditioned only on the pixels in the known region. This approach for image inpainting minimizes the need of post-processing substantially.

## 3. Datasets Used

The following datasets have been used in our implementation. Places2 and the mask dataset were used to train the inpainting network, while the COCO dataset was used for the RCNN.

1) Places2 Dataset: Convolutional neural networks (CNNs) trained on the Places2 Database can be used for scene recognition as well as generic deep scene features for visual recognition [12].
2) Mask Dataset: This is NVIDIA's public train dataset used in Image Inpainting for Irregular Holes Using Partial Convolutions [1].

3) COCO Dataset: COCO is a large-scale object detection, segmentation, and captioning dataset [11].

Figure 2 displays some statistics about these datasets. The MIT Places Dataset is used for input images to train the image inpainting network. For the model to be able to successfully inpaint irregular hole patterns we need images with randomly positioned holes (or masks) having arbitrary shapes. These masks can be manually generated using advanced geometric formulas, but we decided to use NVIDIA's *Irregular Mask Dataset* which contains 55,116 masks for training and 24,866 masks for testing [1], in order to save time. The COCO dataset is used to train the Mask- RCNN.

| Dataset | Images | Categories |
|---|---|---|
| Places2 | ~1.8 million | 365 |
| COCO | ~330K (>200K labeled) | 1.5 million object instances |
| Mask | ~55K training, ~24K testing masks | N/A |

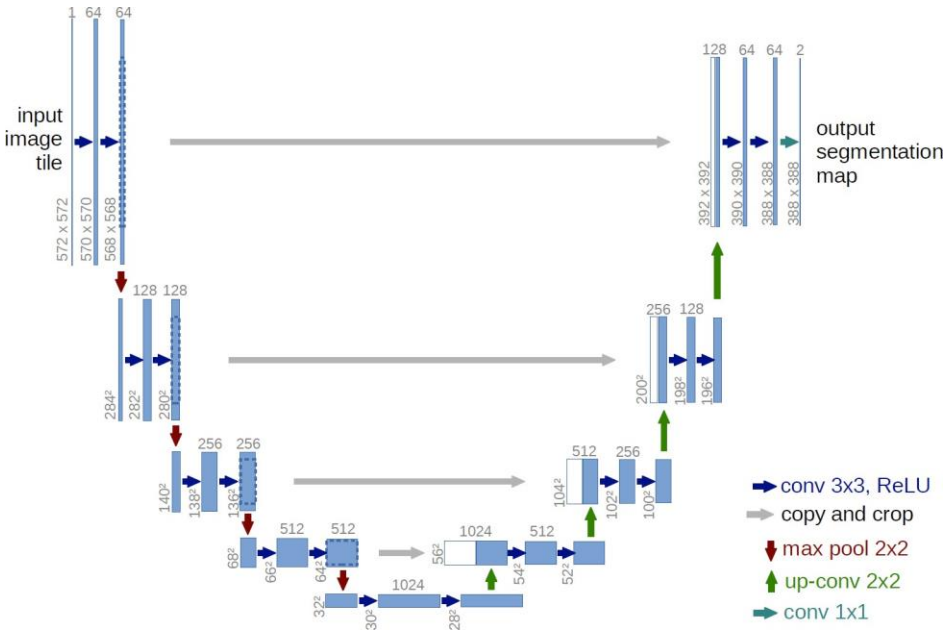
***Figure 2.*** *Dataset statistics.*

## 4. Deep Learning Architecture

This section describes the architecture of the model, the application it addresses, the salient features of the implementation and the key challenges.

### 4.1. Architecture

We decided to use a pix2pix network for this work, like the one used in the publication by Nvidia - "Image inpainting with partial convolutions" (Liu, G. et al.) [1]. It uses a U-Net generic architecture like the one below. We replace every instance of a convolutional layer with a partial convolution. Equation (1) is used in partial convolutions.




Credit: CVG Freiburg

***Figure 3.*** *U-Net architecture (example for 32 x 32 pixels in the lowest resolution). Each blue box corresponds to a multi-channel feature map. The number of channels is denoted on top of the box. The x-y-size is provided at the lower left edge of the box. White boxes represent copied feature maps. The arrows denote the different operations [8].*

The convolution operation involves scanning the entire image using kernels to extract features. In the context of image inpainting, the network must be capable of extracting features from the entire image except the region covered by

the mask. Hence, the desired operation is one which performs the functionality of the convolution operation but is also more selective in the regions from where it extracts features. Due to this reason, convolutional layers are replaced by partial convolution layers.

$$x' = \begin{cases} W^T(X \odot M)\frac{sum(1)}{sum(M)} + b & \text{if sum(M)} > 0 \\ 0, & \text{otherwise} \end{cases} \quad (1)$$

## 4.2. Application

The network receives an image and a mask as input, and outputs a reconstructed image in an undetectable manner. There are a total of 80 object categories to consider and the images are relatively large - which has an impact on the training speed of the network. However, the whole image is used as input to the network. The mask is interpreted as the region of the image that needs to be removed and inpainted. We have decided to process the input as one RGBA image, which means the mask is added as a 4th channel to the image, and represents the "transparency" but can only take values of 1 or 0 (visible or hidden). If the network works properly, the mask should be all ones in the output.

### 4.2.1. Mask R-CNN

A typical CNN can only tell you the class of the objects but not where they are located, but a Region-based-CNN (R-CNN) is capable of detecting the exact location of the objects as well. Mask R-CNN is one of the most popular frameworks for performing instance segmentation. It extends the Faster-R-CNN framework by adding a branch to detect the object mask in parallel. We use the Mask R-CNN to detect the masks for different objects in the image and then this image is fed into the partial convolutional U-Net along with the mask.

### 4.2.2. U-Net Architecture

U-Net was first introduced by Ronneberger et. al. [8], when it showed outstanding results for biomedical image segmentation. Since then, it has become a very popular architecture for image.

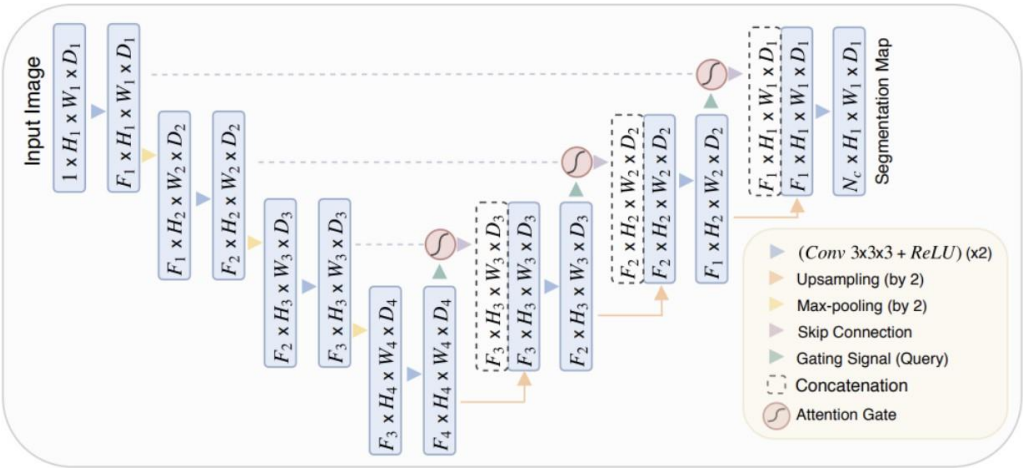



***Figure 4.*** *Lock diagram of the Attention U-Net segmentation model, proposed by Oktay et. al [15].*

segmentation and image to image synthesis. Figure 3 shows the architecture, where a single down-sample layer includes going through a series of two "same" convolutions followed by a max pooling layer. The U-Net architecture has several down-sampling layers where the number of channels is doubled after each one. The layer that comes next is a bottle-neck layer, followed by a series of up-sampling layers. Every up-sampling layer has an incoming skip connection from the corresponding down-sampling layer. The output of the previous layer and the incoming output from the skip connections are concatenated and up sampled using a transposed convolution layer.

The architecture used is particularly unique because of the following reasons-

1) While down sampling the image and mask to encode the important features, partial convolutional layers are used instead of standard convolutions.
2) The mask is added as a separate channel to the input image and hence the network learns to extract features accordingly.
3) The skip connections are applied to both the images and the masks to transfer the features extracted from the

down-sampling to the up-sampling layers.

#### 4.2.3. Modifying U-Net with Attention

As attention mechanisms started getting popular in the deep learning community, attention U-net was introduced by Oktay et. al. [15] to perform image segmentation. It performed better than the regular U-Net. The main idea is to pay attention to the important features in the skip connections and to forget the ones that are not required for segmentation. The type of attention used is gated attention and the architecture is shown in Figure 4.

#### 4.2.4. Partial Convolutions

In our implementation, we use the attention U-net architecture, but we use partial convolutions instead of regular convolutions. The input to our network is the image along with a mask. The mask represents the part of the image that needs to be inpainted. Hence the attention is only applied to the features extracted from the image. This is done because it does not make sense to apply attention to the mask.

## 5. Experiments and Results

Several experiments have been conducted, and various strategies attempted. This section throws light on the most notable developments and accomplished work.

### 5.1. Mask and Partial Convolution

The first step in this research was to modify the convolutional layers in the network by partial convolutions. This is more challenging than it looks, as it also requires modifying the behavior of skip connections, and the way the network processes input in both the encoding and decoding parts. We have extended the parent class Layer in TensorFlow to create the Partial Convolution Layer from scratch.

### 5.2. Loss Implementation

$$L_{hole} = \frac{1}{N_{I_{gt}}} \left\| (1 - M) \odot \left( I_{out} - I_{gt} \right) \right\|_1 \tag{2}$$

$$L_{valid} = \frac{1}{N_{I_{gt}}} \left\| N \odot \left( I_{out} - I_{gt} \right) \right\|_1 \tag{3}$$

$$L_{perceptual} = \sum_{p=0}^{P-1} \frac{\left\| \psi_p^{I_{out}} - \psi_p^{I_{gt}} \right\|_1}{N_{\psi_p I_{gt}}} + \sum_{p=0}^{P-1} \frac{\left\| \psi_p^{I_{comp}} - \psi_p^{I_{gt}} \right\|_1}{N_{\psi_p I_{gt}}} \tag{4}$$

$$L_{style_{out}} = \sum_{p=0}^{P-1} \frac{1}{C_p C_p} \left\| K_p \left( \left( \psi_p^{I_{out}} \right)^T \left( \psi_p^{I_{out}} \right) \right) - \left( \left( \psi_p^{I_{gt}} \right)^T \left( \psi_p^{I_{gt}} \right) \right) \right\|_1 \tag{5}$$

$$L_{style_{comp}} = \sum_{p=0}^{P-1} \frac{1}{C_p C_p} \left\| K_p \left( \left( \psi_p^{I_{comp}} \right)^T \left( \psi_p^{I_{comp}} \right) \right) - \left( \left( \psi_p^{I_{gt}} \right)^T \left( \psi_p^{I_{gt}} \right) \right) \right\|_1 \tag{6}$$

$$L_{tv} = \sum_{(i,j) \in R, (i,j+1) \in R} \frac{\left\| I_{comp}^{i,j+1} - I_{comp}^{i,j} \right\|_1}{N_{I_{comp}}} + \sum_{(i,j) \in R, (i+1,j) \in R} \frac{\left\| I_{comp}^{i+1,j} - I_{comp}^{i,j} \right\|_1}{N_{I_{comp}}} \tag{7}$$

We have implemented all the losses described in the paper as custom loss classes in Tensorflow. There are 6 different kinds of losses implemented which are worth mentioning:

1) Per Pixel Loss for the hidden part of the image (the hole) - mean squared error loss (MSE) (Equation (2))
2) Per Pixel Loss for the visible part of the image - MSE (Equation (3))
3) Perceptual Loss which is an L1 distance in a higher dimensional space, generated after passing the images through a VGG16 to extract the features. This loss improves the smoothness and realism of the output. (Equation (4))
4) Two Style Losses for the output image and the comp image. The comp image is the output image where all the pixels that were not hidden in the input image are set to their original value in post processing (while they may not be in the output image). These style losses are very similar to the one in the original style transfer paper ("A neural algorithm for artistic style" by LA Gatys), which include a Gram matrix transformation before computing the loss. (Equation (5))
5) A Smoothing Loss called the total variation loss, which

expands the hole by one pixel on each side before computing a per pixel difference with the comp (not the image image, see 4). (Equation (6))

6) The final loss is a weighted average of the first five losses (the two style losses are grouped together). (Equation (7))

### 5.3. Object Detection

The eventual goal of our experiment is to select a detected object in an image and remove the object in an undetectable manner. There are two important steps which must be taken to accomplish this -

*1. Detect an object in an image 2. Find the mask for that object*

This mask will serve as the "hole" that is fed into the network along with the image. To do this, we have used a Mask R-CNN [9, 10] model trained on the MS COCO dataset [11]. Mask R-CNN is a popular model that performs instance segmentation. There are three outputs for each candidate object - 1. Class Label, 2. Bounding Box, 3. Object Mask, as show in Figure 5. This object mask will be used as the "hole" along with the original image for the image inpainting network.


(a) Original Image (b) Generated boxes

***Figure 5.*** *R-CNN output example [16].*

### 5.4. Results

We tried running our system on various images. We have included some of the results along with their respective masks in Appendix A.

### 5.5. Challenges

Although we managed to implement all the different parts of the model, we could not sufficiently train it. We expect the model to need at least 10 days of training and 1 or 2 days of fine tuning to achieve results similar in quality to the Nvidia publication.

Rather than just waiting for the model to train, we decided to use pre-trained weights and feed them into our model. This way, we were able to test the whole pipeline and see if our original idea of combining inpainting with R-CNN was credible.

## 6. Conclusion

Image inpainting with deep learning can reduce post processing expenses. It finds applications in various image editing tools. The system we have proposed can be incorporated as a user-friendly feature in any of these tools. Removing unwanted objects from an otherwise perfect image will now be as easy as clicking on the objects to make them disappear. Our system eliminates the need of painstakingly painting regions in the image to be removed. It is also an improvement on tools that need you to manually draw a box around the object to be removed.

The first step in this work was to modify the convolutional layers in the network by partial convolutions. This required us to modify the behavior of the skip connections and the way the network processed inputs in U-Net's encoding and decoding parts. To do this, we created the Partial Convolution Layer from scratch using TensorFlow. Six of the losses described in Nvidia's paper were implemented and it was interesting to see how they interacted together. We tried modifying each of their coefficients and observed that the per pixel loss and the style loss had the biggest impact on the results, while the total variation and perceptual losses were mostly for fine tuning. We also realized how the capabilities of a CNN can be extended. A typical CNN can only tell you the class of the objects but not where they are located, whereas an R-CNN can detect the exact location of the objects as well. The novelty in our system lies in the inclusion of object detection to supplement the image inpainting component.

The system we built over the course of this research produced reasonable results under the given constraints, as discussed in the previous sections. However, there is room for improvement. We did face a degradation in the quality of the final image, which we suspect is an issue with our image

normalization parameters. Also, the inpainted region did not always seamlessly blend in with the surroundings. Certainly, our model needs to be trained for a lot longer to resolve this problem. Our approach aligns with the principles of cost-effective design and modeling, as described in [17], which demonstrated efficient object detection and image inpainting using low-budget hardware. Future iterations of our system could further optimize computational efficiency and model refinement to achieve high performance while maintaining frugality in resource utilization.

However, the system also has some limitations. It struggles to remove an object which is overlapping with other objects. Also, feeding the system with an image that is cluttered with a bunch of objects does not give us the desired results. Another point to note is that our system is only as good as its object detection component, which is responsible for producing a masked version of the image, without the selected object to be removed. This can prove to be the system's single point of failure.

Nonetheless, we gained a significant amount of knowledge in the realm of image processing and manipulation using deep learning. We learnt about object detection, partial convolutions and its advantages over regular convolutions in this area, the U-Net architecture, various losses and how they influence the objective of a model and how to incorporate attention in the U-Net architecture. Although the results fell short of our expectations, we gained invaluable experience with powerful deep learning concepts over the course of this work.

## 7. Future Work

The pervasiveness of neural and large language models has now extended into prior technological fields such as IoT, finding practical applications in domains like smart infrastructure and resource optimization [18]. Future work could explore leveraging structural priors and fine-tuned embeddings, similar to the experimental approach used in [19], to improve the integration of object detection and image inpainting for more seamless reconstructions.

Firstly, we intend to work on improving the drawbacks of our current approach. To start with, our focus would be on making training more efficient to push the capability of our model while requiring shorter compute time. As mentioned above, we use pre-trained weights to arrive at our results, but we intend to address this as well, and switch to a model that is trained by us from scratch - thus providing more flexibility in our approach. The object detection component of the system allows a user to choose or highlight the elements of a class instead of drawing a box around it on the actual image. This has the potential to allow a much cleaner and more optimal inpainting. It also mitigates the chance of human error. However, the implementation of this is not fully stable yet and must be enhanced further. We also plan to increase the number of classes and create different CNNs to deal with different class groups such as animals, species, vehicles and so forth.

Secondly, there are several possible ways to extend the proposed product, by considering existing technology as well as other prospective applications. Once the model has reached a mature state, some of these extensions will be considered. We are thinking about extending the system to apply it to videos, while keeping the image consistent from one frame to the next. To do this, we would need to add a temporal loss to the six existing losses to ensure this consistency. Another noteworthy extension of the product would be to enable an option to replace an object of one class with an object of another class, as opposed to the primary function of just removing an object from an image. Lastly, an interesting although somewhat divergent application would be to supplement a similar system to work with audio clips (for example, by specifying it to remove a specific instrument from an audio track). Along those lines, future work could explore multimodal datasets, such as the AudioFacialMatrix dataset [20], to enhance object detection and image inpainting by integrating complementary data sources. Leveraging diverse modalities may improve model robustness and generalization across varied real-world scenarios with potential extensions to diverse applications such as recommender systems [21, 22].

## Abbreviations

| | |
|---|---|
| AI | Artificial Intelligence |
| CNN | Convolutional Neural Network |
| R-CNN | Region-based CNN |
| U-Net | U-shaped Convolutional Neural Network |
| MSE | Mean Squared Error |
| VGG-16 | Visual Geometry Group 16-layer CNN |
| COCO | Common Objects in Context dataset |
| RGBA | Red, Green, Blue, Alpha |
| IoT | Internet of Things |

## Acknowledgments

We would like to thank the deep learning community at Carnegie Mellon University for their constant support during this work. Additionally, we want to express our gratitude to Professor Bhiksha Raj, for providing helpful suggestions and feedback. We also acknowledge the contributions of the open-source and research communities whose work laid the foundation for our study. Lastly, we appreciate the feedback and encouragement from our peers, reviewers, and collaborators, whose thoughtful suggestions have helped improve the quality of this paper.

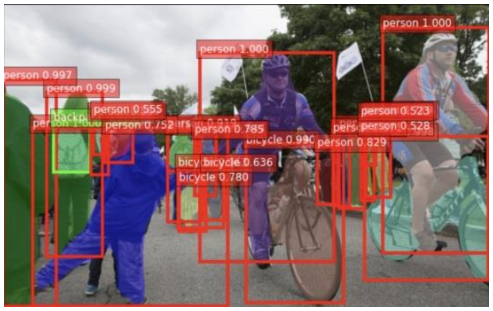
## Conflicts of Interest

The authors declare no conflict of interest.

# Appendix

*A End-to-end system results*

(a) Original Image




(b) Objects detected by R-CNN

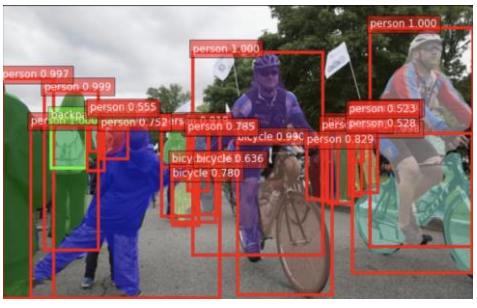
(c) Selected object mask

(d) Image with mask applied

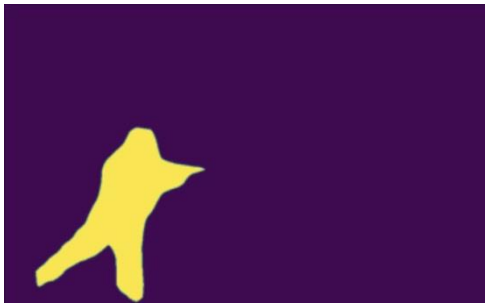
(e) Final Inpainted Image

***Figure 6.*** *End-to-end system results - 1.*

(a) Original Image




(b) Objects detected by R-CNN



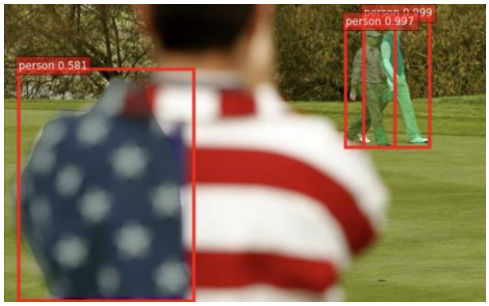
(c) Selected object mask

(d) Image with mask applied

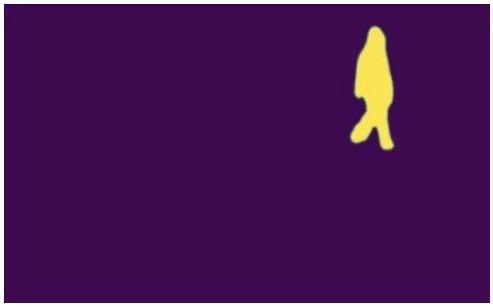
(e) Final Inpainted Image

***Figure 7.*** *End-to-end system results - 2.*

(a) Original Image




(b) Objects detected by R-CNN



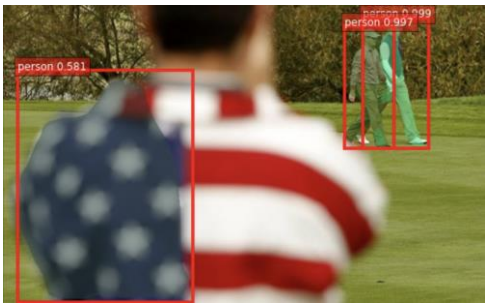
(c) Selected object mask

(d) Image with mask applied

(e) Final Inpainted Image

***Figure 8.*** *End-to-end system results - 3.*

(a) Original Image




(b) Objects detected by R-CNN

(c) Selected object mask

(d) Image with mask applied

(e) Final Inpainted Image

***Figure 9.*** *End-to-end system results - 4.*

## References


[1] Liu, G., Tao, T. W. A., Catanzaro, B. (2018) Image Inpainting for Irregular Holes Using Partial Convolutions. Computer Vision and Pattern Recognition (cs. CV).

[2] Sun, J., Yuan, L., Jia, J., Shum, H. (2005) Image Completion with Structure Propagation. ACM Transactions on Graphics. https://doi.org/10.1145/1186822.1073274

[3] Pearl, J. (1988) Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference, Morgan Kaufmann Publishers, Inc.

[4] Barnes, C., Shechtman, E., Finkelstein, A., Goldman, D. B. (2009) Patchmatch: A randomized correspondence algorithm for structural image editing. ACM Transactions on Graphics-TOG.

[5] Köhler R., Schuler C., Schölkopf B., Harmeling S. (2014) Mask-Specific Inpainting with Deep Neural Networks. Pattern Recognition. GCPR 2014. Lecture Notes in Computer Science, vol 8753. Springer, Cham. https://doi.org/10.1007/978-3-319-11752-2_43

[6] Yeh, R. A., Chen, C., Yian Lim, T., Schwing, A. G., Hasegawa-Johnson, M., & Do, M. N. (2017) Semantic image inpainting with deep generative models. 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, pp. 6882-6890. https://doi.org/10.1109/CVPR.2017.728

[7] Ulyanov, D., Vedaldi, A., Lempitsky, V. (2017), Deep Image Prior, arXiv: 1711.10925 [cs. CV].

[8] Ronneberger O., Fischer P., Brox T. (2015) U-Net: Convolutional Networks for Biomedical Image Segmentation. In: Navab N., Hornegger J., Wells W., Frangi A. (eds), Medical Image Computing and Computer-Assisted Intervention - MICCAI 2015, vol 9351. https://doi.org/10.48550/arXiv.1505.04597

[9] Girshick, R., Donahue, J., Darrell, T., Malik, J. (2014) Rich feature hierarchies for accurate object detection and semantic segmentation, arXiv: 1311.2524 [cs. CV]. https://doi.org/10.48550/arXiv.1311.2524

[10] He, K., Gkioxari, G., Dollár, P., Girshick, R. (2018) Mask R-CNN, arXiv: 1703.06870 [cs. CV].

[11] Lin TY. et al. (2014) Microsoft COCO: Common Objects in Context. In: Fleet D., Pajdla T., Schiele B., Tuytelaars T. (eds) Computer Vision - ECCV 2014, vol 8693. Springer, Cham.

[12] Zhou, B., Khosla, A., Lapedriza, A., Torralba, A., Oliva, A. Places: A 10 million Image Database for Scene Recognition, arXiv:1610.02055. https://doi.org/10.48550/arXiv.1610.02055

[13] Liu, Z. & Luo, P. & Wang, X.& Tang, X. (2018) Deep Learning Face Attributes in the Wild. Proceedings of International Conference on Computer Vision (ICCV).

[14] Singh, R., Bodhe, A., Kanuparthi, P., Ananthakrishnan, A. and Pitt, J. (2019) From classification to definition: The changing nature of human adjudication, IEEE Technol. Soc. Mag., vol. 38, no. 4, pp. 55-62. https://doi.org/10.1109/MTS.2019.2948441

[15] Oktay, O., Schlemper, J., Folgoc, L., Lee, M., Heinrich, M., Misawa, K., Mori, K., McDonagh, S., Hammerla, N. Y., Kainz, B., Glocker, B., Rueckert, D. (2018) Attention U-Net: Learning Where to Look for the Pancreas, arXiv: 1804.03999 [cs. CV]. https://doi.org/10.48550/arXiv.1804.03999

[16] Singh, R., Bruder, M., Joshi, R., Kumar, A. An Adversarial Approach to Image Inpainting. GitHub repository, https://github.com/RealBrionac/PartialConv_Tensorflow

[17] Singh, R. (2023) Cost Optimization of Neural Models for Inpainting, ResearchGate, press.

[18] Bodhe, A. R., Singh, R., and Bawa, A. (2016) An Internet of Things solution for sustainable domestic water consumption, International Conference on Computation System and Information Technology for Sustainable Solutions (CSITSS), Bengaluru, India, pp. 224-229.

[19] Umapathy, A., Radhakrishnan, K., Jain, K., and Singh, R. (2020) CiteQA@CLSciSumm 2020. In Proceedings of the First Workshop on Scholarly Document Processing, Association for Computational Linguistics, pp. 297-302. https://doi.org/10.18653/v1/2020.sdp-1.34

[20] Singh, R. (2024) AudioFacialMatrix: Dataset for Voice and Face AI, 8th CSITSS IEEE, Bengaluru, India, pp. 1-5. https://doi.org/10.1109/CSITSS64042.2024.10816934

[21] Singh, R. (2024) The Recommender System Paradox: Autonomy in the Age of AI Recommendations, 2024 3rd International Joint Conference on Information and Communication Engineering (JCICE), pp. 139-143. https://doi.org/10.1109/JCICE61382.2024.00037

[22] Singh, R. (2023) Related Rhythms: Recommendation System To Discover Music You May Like, arXiv: 2309.13544 [cs. IR]. https://doi.org/10.48550/arXiv.2309.13544