Better Image De-occlusion Using Similar Images

Team #8
Guoyuan An (20184637), Saelyne Yang (20193338),
Heeju Wi (20203388), Junyong Park (20205146)

Introduction

Image De-occlusion

Image de-occlusion aims to recover and complete occluded parts of an image

(a) Original image

(b) Image de-occlusion

(not actual de-occlusion,
just an example to show the concept)

Image De-occlusion

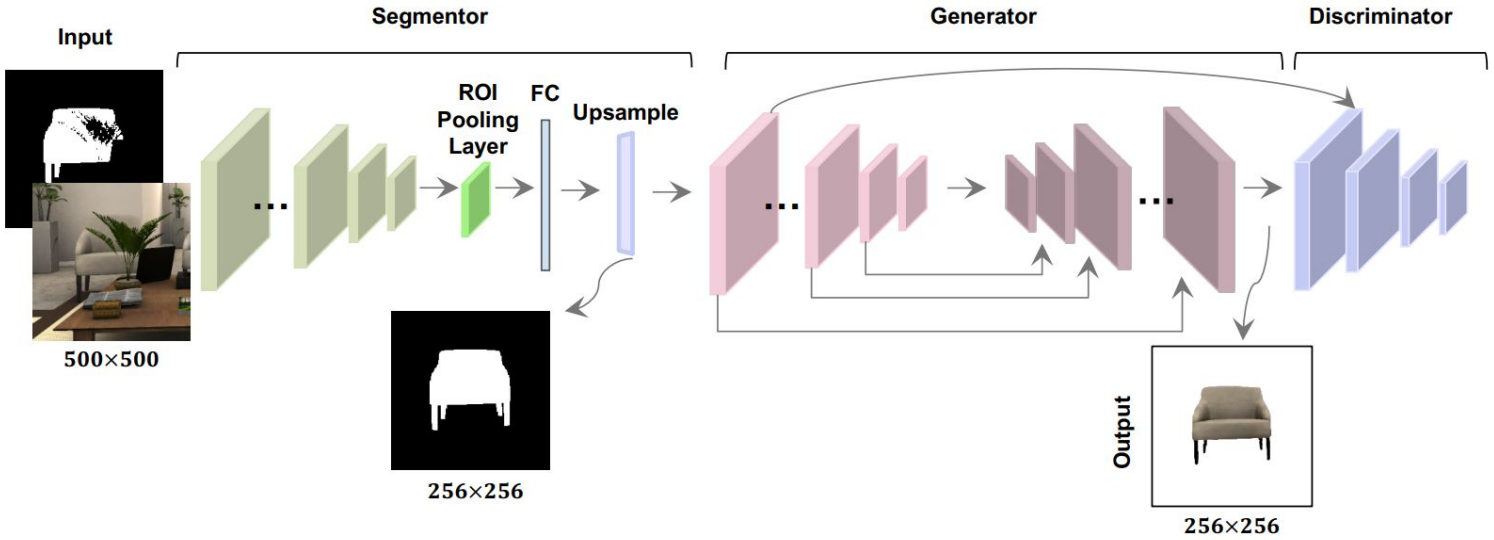
General process of Image de-occlusion

Input image | Modal mask | Amodal mask | De-occlusion

Image segmentation → Masking invisible parts → Filling in colors

Related work on Image De-occlusion

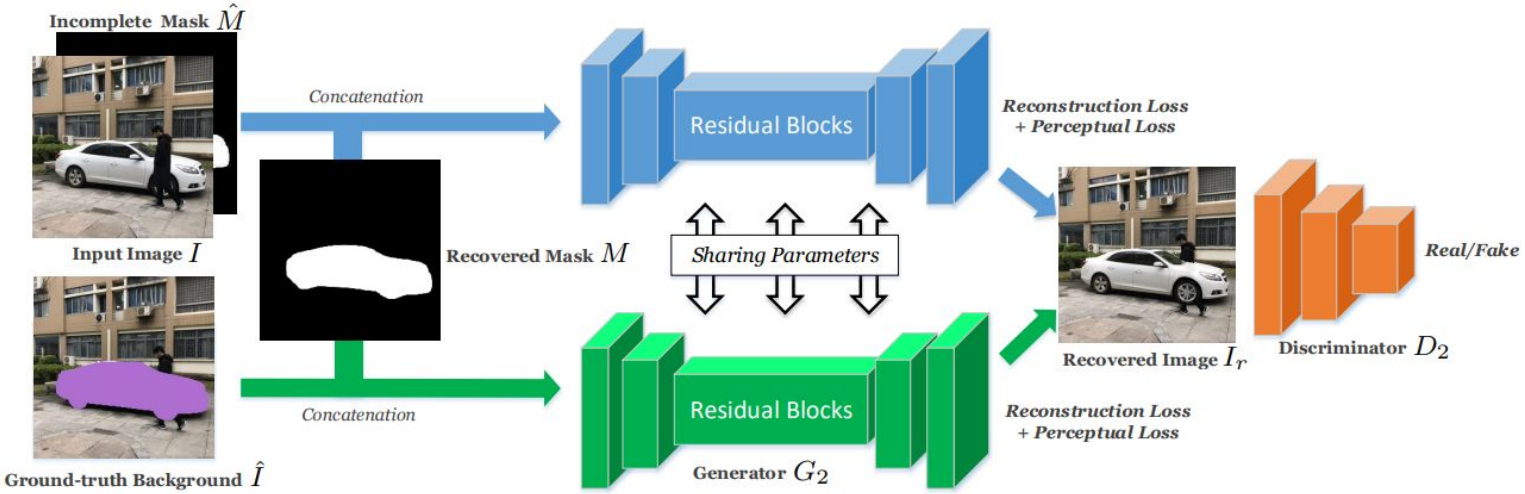
1. SeGAN: Segmenting and Generating the Invisible (Ehsani et al., CVPR 2018)



With the RGB image and the segmentation mask,
they used **GAN-like model** to reconstruct RGB of the occluded regions of the object

Related work on Image De-occlusion

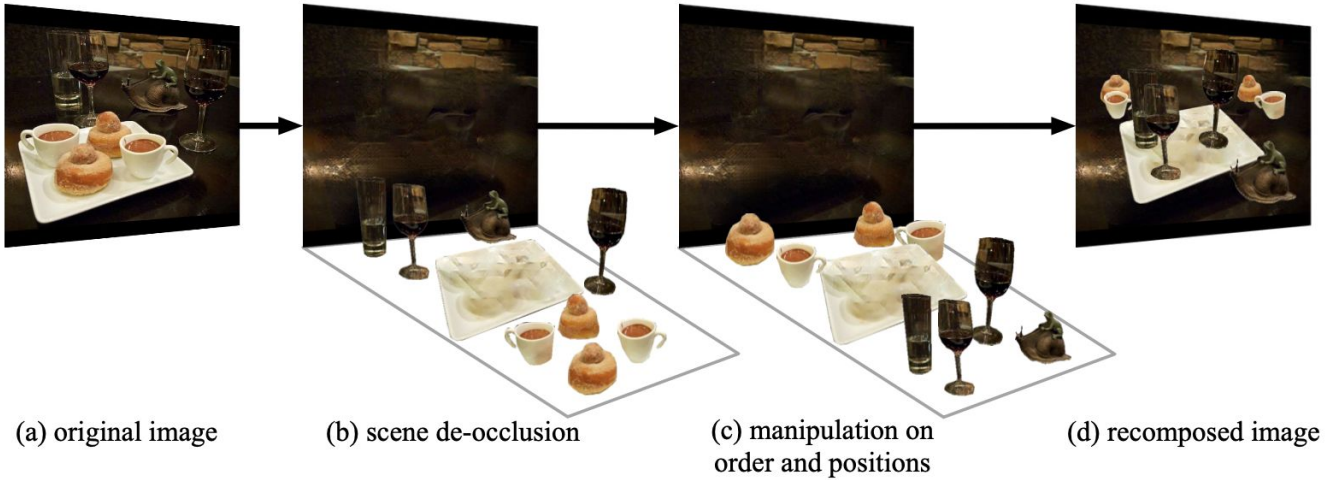
2. Visualizing the Invisible: Occluded Vehicle Segmentation and Recovery (Yan et al., ICCV 2019)



They also used **GAN-like model** consisting of **Segmentation completion** and **Appearance recovery**

Replicated paper

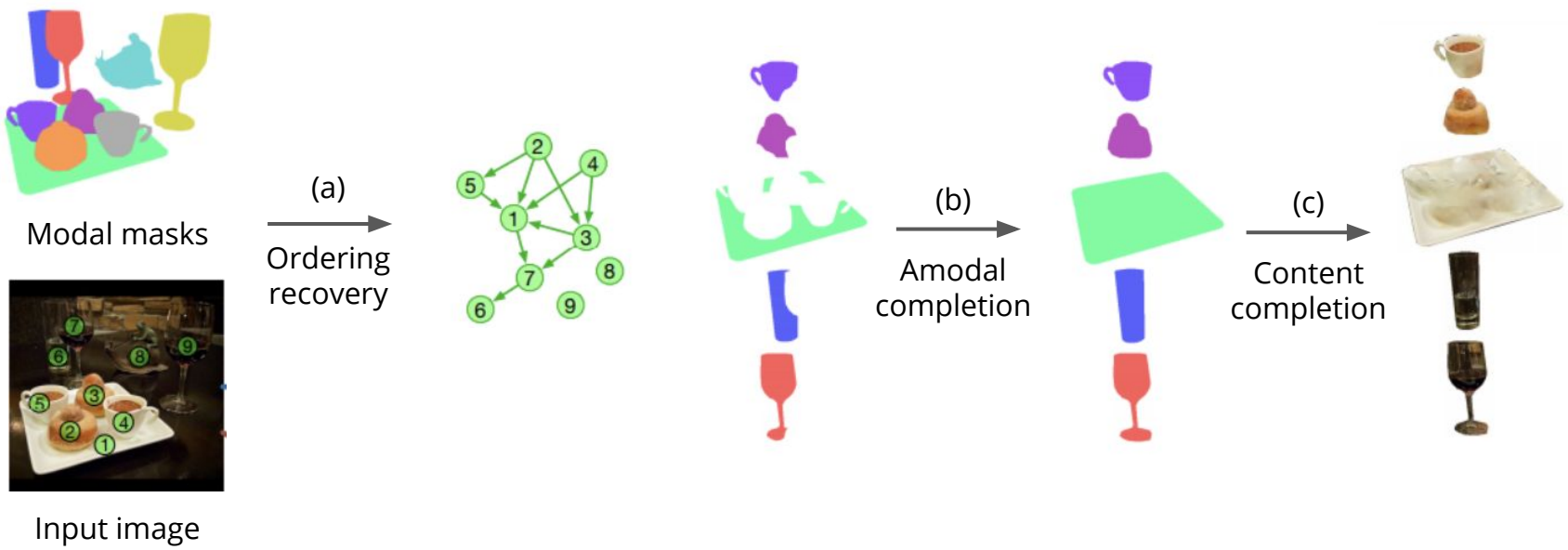
Self-Supervised Scene De-occlusion (Zhan et al., CVPR 2020)



Self-supervised method of training scene de-occlusion
by erasing random parts of images

Replicated paper

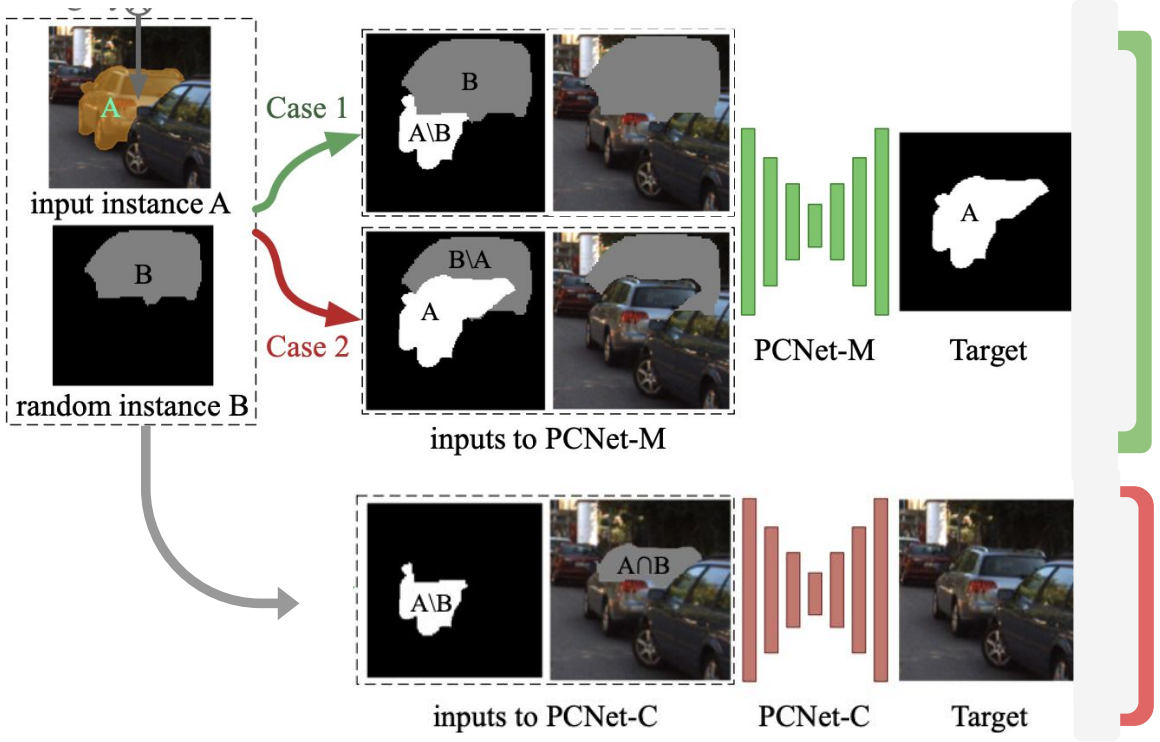
Self-Supervised Scene De-occlusion (Zhan et al., CVPR 2020)



Overview of the proposed framework

Replicated paper

Self-Supervised Scene De-occlusion (Zhan et al., CVPR 2020)



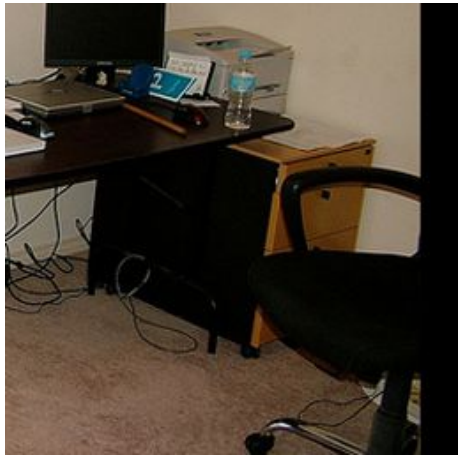
(a) PCNet-M
Mask Completion

(b) PCNet-C
Content Completion

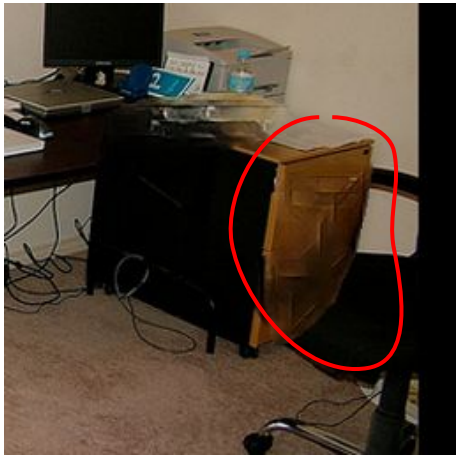
Limitation of the replicated paper

Self-Supervised Scene De-occlusion (Zhan et al., CVPR 2020)

(a) Original image



(b) De-occlusion result

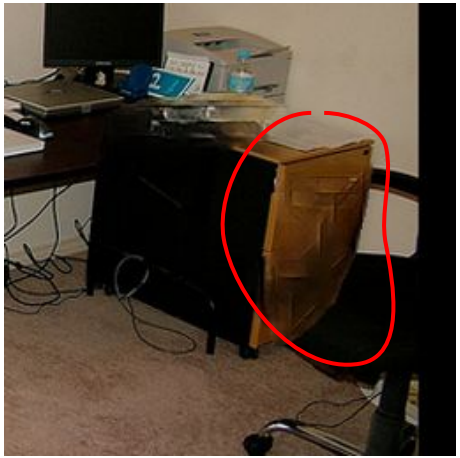


Quality of de-occlusion is still not very accurate;
It **doesn't clearly remove occluding objects** when filing the content that was occluded.

Improvement approach

Improvement approach

How could we improve the quality of image de-occlusion?



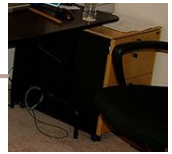
- There still remains **traces of occluding objects** in current de-occlusion results
- How to providing visual information of the target object?
  → Use **similar images** to a given image!

e.g. ‘drawer’

visual information

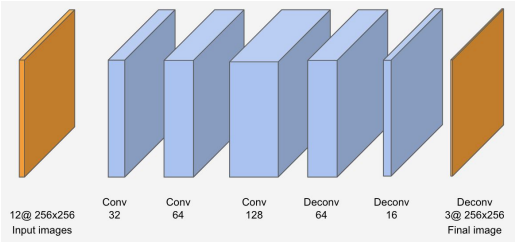
Improvement approach

Similar images-based approach



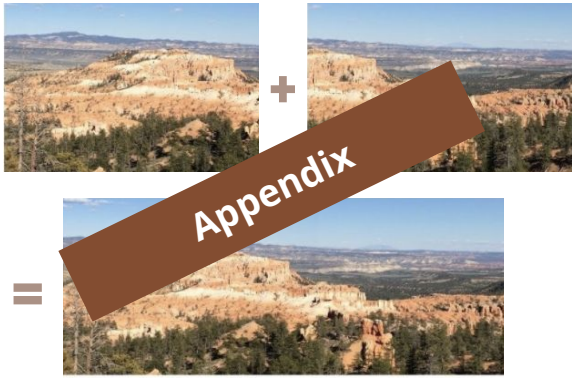
Reference images
(i.e. similar images)

1) New convolutional model



incorporating reference images along with the outcomes from the existing model

2) Image Stitching



combining multiple images with overlapping fields of view

Dataset

Dataset

1) COCOA dataset[1]

- Subset of COCO dataset[2]
- Dataset used in our **replicated paper**
- **Modal and Amodal** masks, Base image
- Training data: 2500 images, 22163 instances
- Testing data: 1323 images, 12753 instances

[1] Zhuh et al., 2015, Semantic amodal segmentation
[2] Lin et al., 2014, Microsoft COCO: common objects in context

Dataset

2) Our new dataset (modified COCOA dataset)

sample data

Erased image

Predicted image
from the original model

Reference image
(similar image)

Dataset attributes

- Modal mask, Amodal mask, Base image
  - *original COCOA dataset*
- *new!* Predicted amodal mask, Predicted image
  - *results from the model used in the replicated paper*
- *new!* Eraser, Erased mask
  - *artificially erased some part of the given image for testing*
- *new!* **Reference image**
  - *similar images retrieved by ourselves*

Dataset size

- **9440** objects out of 22163 in original COCOA dataset

Experiments

Experiments - 1) Image Search API

Implemented an Image Search API that searches for images by a given image

Target image

Host images on
imgbb.com

Google search by image

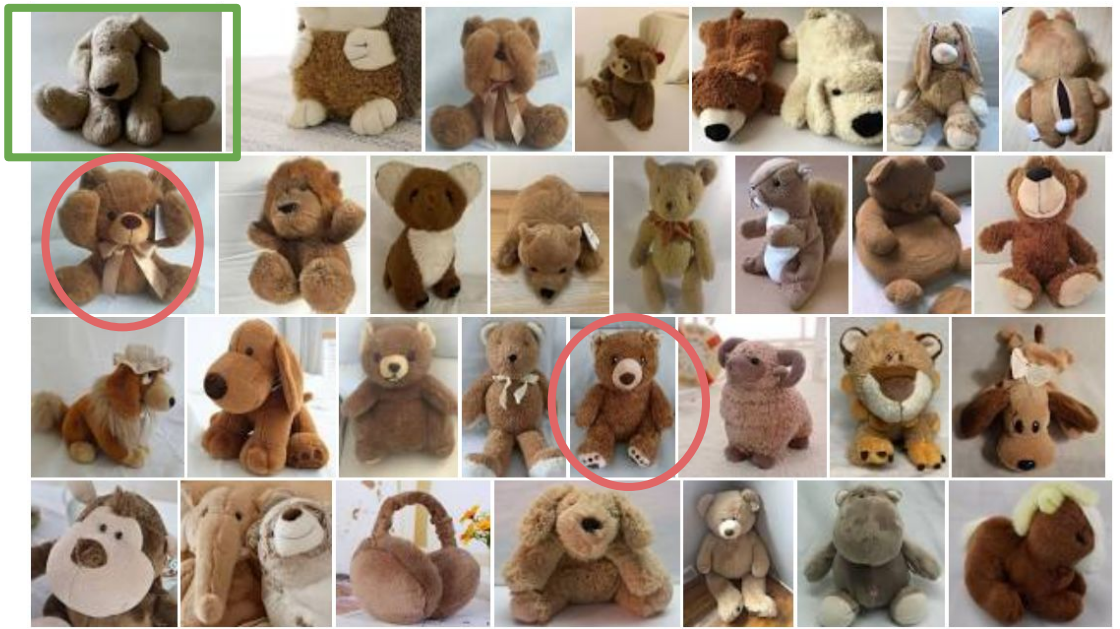
Get images

Automatically searches for images with a series of given image on Google

Code released on github.com/jyp0802/get_reference_images/blob/master/search.py

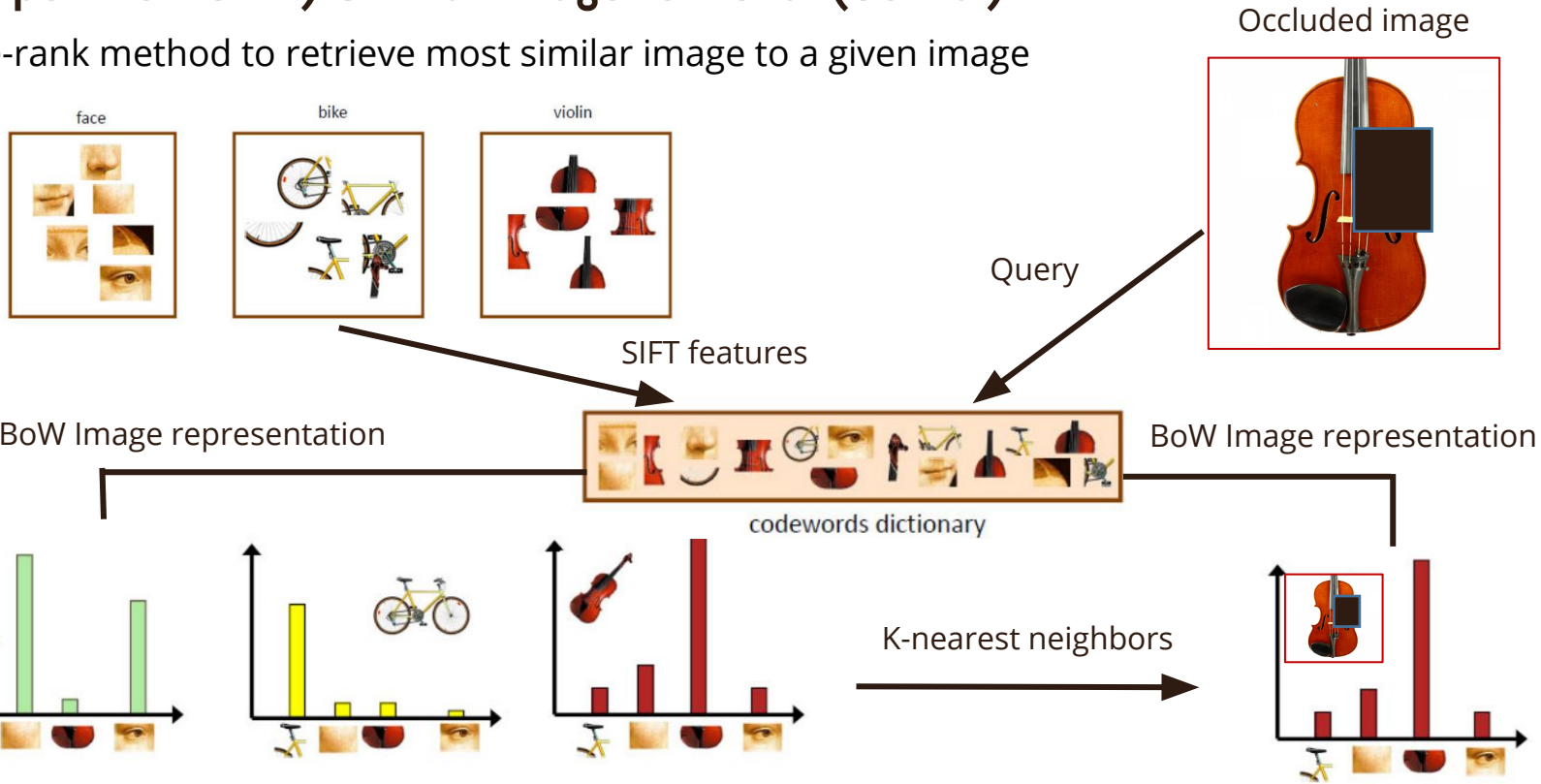
Experiments - 2) Similar image retrieval

Re-rank from the Google search result



Experiments - 2) Similar image retrieval (Contd.)

Re-rank method to retrieve most similar image to a given image



Experiments - 3) Our new convolutional model

A new convolutional model with our modified dataset

Inputs to the model

result from our replication paper

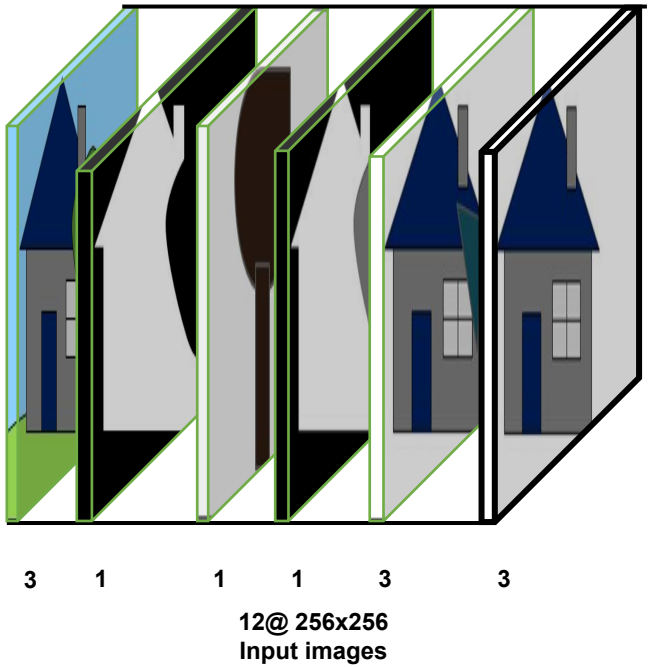
1) *occluded **RGB** image of the target object*
2) *modal mask of the target object*
3) *mask of the occluding objects*
4) *predicted amodal mask of the target object (from PCNet-M)*
5) *predicted de-occluded **RGB** image (from PCNet-C)*

new data

6) *reference image from image search (**RGB**) (training & testing)*

+

6) *random transformation of ground truth image (**RGB**) (Only for training)*



Experiments - 3) Our new convolutional model (Cont.)

Transform the ground truth images to get more reference images.

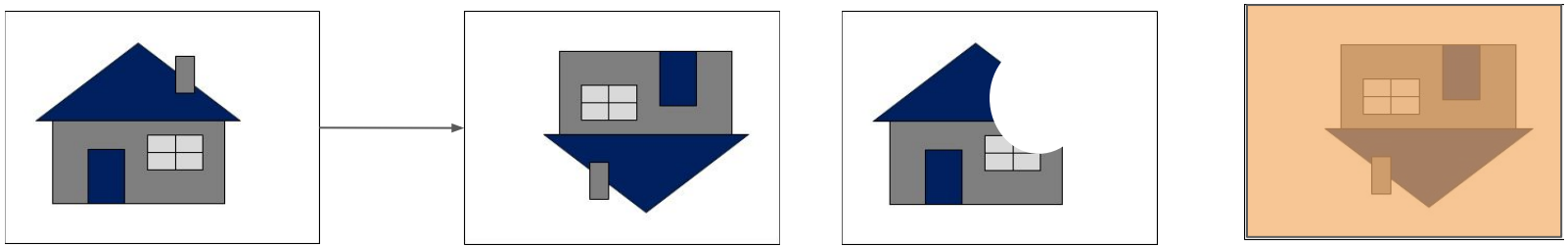
new data

*6) reference image from image search (**RGB**) (training & testing)*

+

*6) random transformation of ground truth image (**RGB**) (Only for training)*

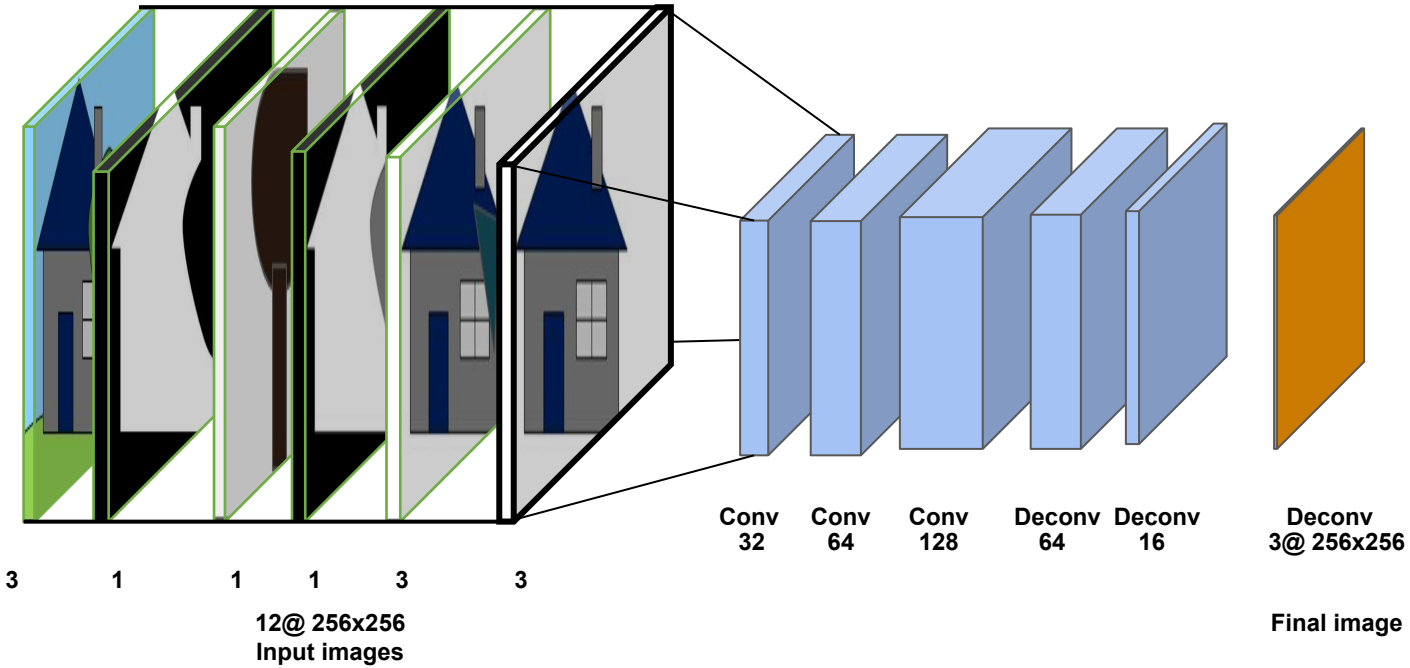
Detail of transformation



origin *rotation* *Crop* *Gaussian Noise*

Experiments - 3) Our new convolutional model (Cont.)

A new convolutional model with our modified dataset



Experiments - 3) Our new convolutional model

Experiment setting and Environment

- Environment
  - Implemented with PyTorch
  - Training took around 6 hours with a RTX 2080Ti GPU.

- Experiment setting
  - epochs = 100
  - batch size = 16
  - learning rate = 0.001
  - optimizer = Stochastic gradient
  - loss function = mean squared error on the pixels of the occluded area that needs to be filled

```
126
127  def train():
128      total_step = len(trainloader)
129      for epoch in range(num_epochs):
130          for i, inputs in enumerate(trainloader):
131              for idx in range(len(inputs)):
132                  inputs[idx] = inputs[idx].to(device)
133
134              rgb_occ, mod_mask, amod_mask, rgb_pred, ref, rgb_gt = inputs
135
136              imgs_in = torch.cat(inputs[:-1], 1)
137
138              # Forward pass
139              rgb_out = model(imgs_in).to(device)
140              loss = criterion(rgb_out, rgb_gt)
141
142              # Backward and optimize
143              optimizer.zero_grad()
144              loss.backward()
145              optimizer.step()
146
147              if (epoch+1) % 1 == 0:
148                  print ('Epoch [{}/{}], Step [{}/{}], Loss: {:.4f}'
149                         .format(epoch+1, num_epochs, i+1, total_step, loss.
```

Code available on
github.com/jyp0802/get_reference_images/blob/master/refineimage.py

Results

Results - 1) Replication results

Our replicated model performance is **similar** to the original model , especially on **COCOA** dataset

MODEL / DATASET	COCOA	KINS
Original model	87.1	92.5
Replicated model	87.112	88.803

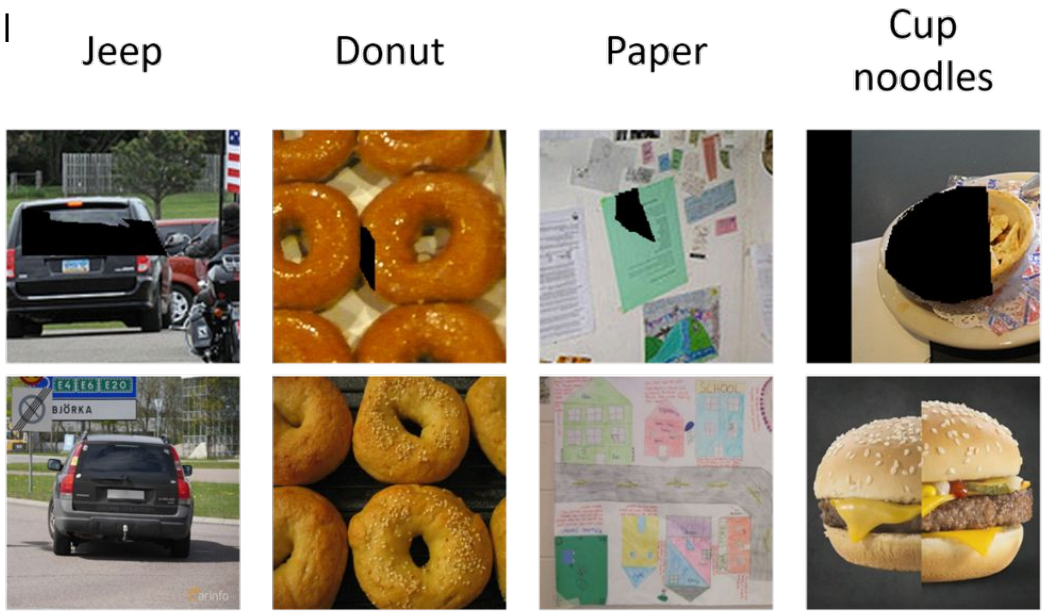
[Table 1] Pair-wise accuracy on occluded instance pairs

MODEL / DATASET	COCOA (%MIoU)	KINS (%MIoU)
Original model	81.35	94.76
Replicated model	81.346	94.356

[Table 2] Amodal completion results (MIoU = Mean intersection over Union)

Results - 2) Our improved model results

Reference image retrieval



Good instances of reference images

Similar image but not helpful

Doesn't work well if occlusion is too large

Results - 2) Our improved model results

Image De-occlusion: Quantitative Evaluation of model output

Model	Euclidean	Manhattan	Chord	MSE
Original	87.936	118.936	0.481	87.937
Altered-GT	94.630	**117.460**	**0.371**	94.630
Similar-Ref	90.961	138.063	**0.357**	90.961

[Table 3] Content completion results on artificially occluded data

Altered-GT: During training, reference images = **transformed ground truth image**
Similar-Ref: During training, reference images = **images retrieved from image search API**

Results - 2) Our improved model results

Image De-occlusion: Qualitative Evaluation of model output

Compared to the PCNet-C, our improved model erases the occluding objects **more clearly**

* Some reference images were chosen by ourselves

Discussion

Discussion - Our approach

- Our convolutional model performed image de-occlusion **slightly more accurately** than the original replicated PCNet-C model

- However, our model is influenced by the **reference images**
  → If the reference image is inaccurate, our model performed poorly.

Discussion - Limitations and Potential improvements

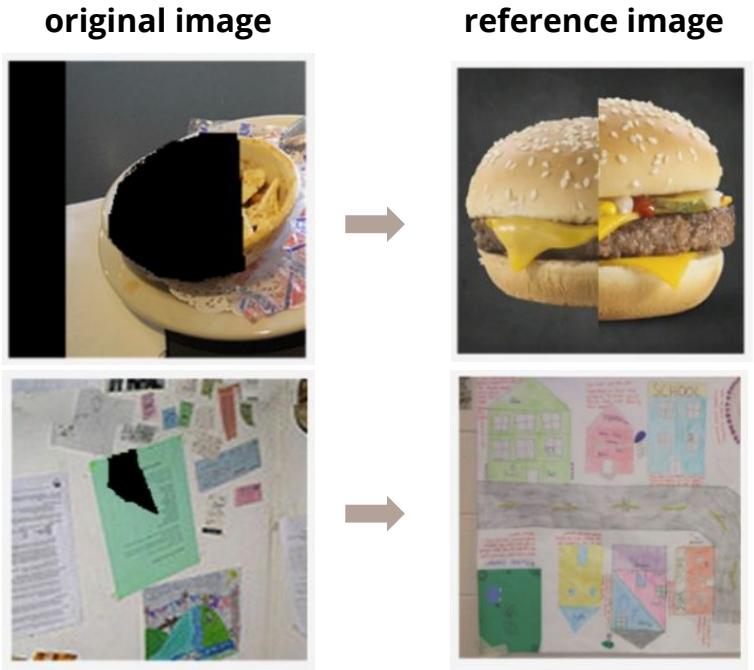
1. Inaccurate reference images

Limitations

- Large occlusion -> incorrect objects
- Unimportant images

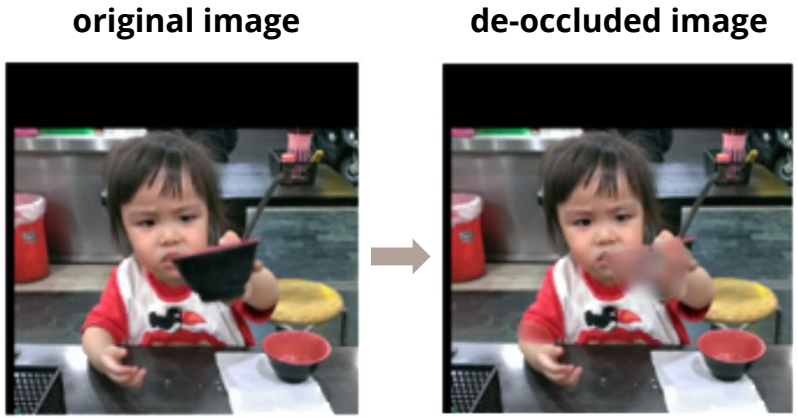
Potential improvements

- Detect the type of the target object
- Make target object focus of the query image



Discussion - Limitations and Potential improvements

2. Poor de-occlusion



Limitations

- Can’t fill in small details
- Final image unnatural

Potential improvements

- Train on a larger dataset with more accurate reference images
- Change to a GAN structure with a discriminator

THANK YOU

Better Image De-occlusion Using Similar Images

Appendix - Image Stitching

Dataset

1) Modified UKBenchmark dataset [1]

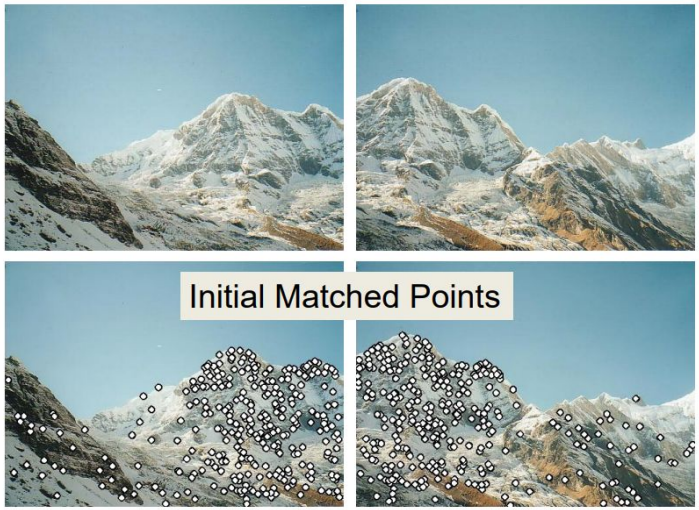
original images occluded images

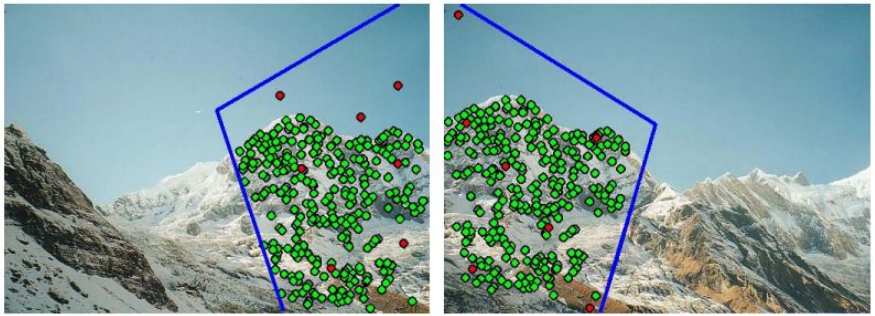
- UKBenchmark: 2550 categories with 4 images of the same object each
- **Cropped** all images so that we can treat as **occluded images**
- Training data: 7650 occluded images
- Testing data: 2550 occluded images

[1] https://archive.org/details/ukbench

Appendix - Image Stitching

Experiment





Appendix - Image Stitching

Results

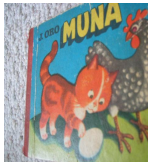
Original images



Occluded images



Reference images





Completion result