# Learning Object Placement by Inpainting for Compositional Data Augmentation

Anonymous ECCV submission

Paper ID 1973

**Abstract.** We study the problem of common sense placement of visual objects in an image. This involves multiple aspects of visual recognition: the instance segmentation of the scene, 3D layout, and common knowledge of how objects are placed and where objects are moving in the 3D scene. This seemingly simple task is difficult for current learning-based approaches because of the lack of labeled training pair of foreground objects paired with cleaned background scenes. We propose a self-learning framework that automatically generates the necessary training data without any manual labeling by detecting, cutting, and inpainting objects from an image. We propose a PlaceNet that predicts a diverse distribution of common sense locations when given a foreground object and a background scene. We show one practical use of our object placement network for augmenting training datasets by recomposition of object-scene with a key property of contextual relationship preservation. We demonstrate improvement of object detection and instance segmentation performance on both Cityscape [4] and KITTI [9] datasets. We also show that the learned representation of our PlaceNet displays strong discriminative power in image retrieval and classification.

**Keywords:** Object Placement, Inpainting, Data Augmentation

## 1 Introduction

Studies in humans and animals suggest that the mental replay of past experiences is essential for enhancing visual procession as well as making action decisions [3]. We ask the question: can developing a computational mental replay model help to improve AI visual perception tasks such as recognition and segmentation? More specifically, would the mental replay of object placement and scene affordance boost visual recognition systems?

This is not only a scientific question, but also a highly practical one for training a deep learning network. Most AI systems based on deep learning have a large appetite for a vast quantity of human-labeled training dataset, and all modern deep learning based algorithms implicitly use contextual cue for recognition tasks. Several recent works demonstrated 'copy-paste' like data augmentation by inserting objects into a background image in order to boost object recognition performance [5][7][6][10][32]. If the mental replay of object placement could be carried out reliably with the preservation of contextual relationship, this method

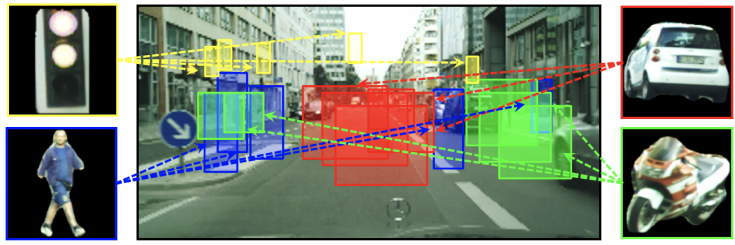

Fig. 1: Given a foreground object and a background scene, we aim to learn a set of reasonable and diverse locations and scales to insert the object into the scene.

leads to a new way of data augmentation by utilizing self-supervised learning of object placement.

Motivated by vast amount of driving scenes in public datasets, we create a self-supervised mental replay task of learning object placements into street scenes. Our system starts by observing many street scene images along with instance segmentation labels. It learns to mental replay: transferring objects from one scene and composite them into other plausible scenes at plausible new locations. This task has many useful side-effects: 1) it encourages the algorithm to discover functionality based object and scene features, and their contextual dependency; 2) it helps to create new object-scene compositions that could potentially balance out biases or augment hard examples in the training dataset.

The self-learning can also come for 'free' just by observing unlabeled scenes. Our insight is that we can generate 'free' labeled training data using an instance segmentation network [11] to cut out objects and fill in the holes using an image inpainting network [33]. The 'free' labeled object-background pairs tell us *what* the object looks like and *where* it is placed.

The 'free' labeled object-background pairs are then fed into our proposed PlaceNet, which predicts the location and scale to insert the object into the background. The key challenge is to learn diverse yet plausible object placements. There is a many-to-many mapping between the objects/scenes with plausible placement solutions. For example, one object-scene image pair can correspond to many different object placements (one-to-many). At the same time, similar object-scene pairs can correspond to the same object placement (many-to-one). The two key properties we want are 1) *diversity*: learns a many-to-many mapping, where images consisting of similar object-scene pairs can share the similar distributions of solutions; and 2) *modularity*: the objects and scenes are represented modularly to allow for maximal composition possibility for inserting objects into scenes.

We demonstrate that our PlaceNet can outperform strong baselines in terms of plausibility and diversity in object placement learning. In addition, we show two useful applications of our object placement learning. First, we use the learned PlaceNet to insert objects from one scene into many other scenes with natural object-context relationship in order to augment training data for boosting object detection and instance segmentation. Our hypothesis is that by compositing scenes that model the distribution of any object, we are able to improve the de-

tection and segmentation performance by allowing the detectors [27][11] see more object-context relationships. Second, we show that our self-learning PlaceNet can learn meaningful features for object/scene retrieval as well as image classification.

## 2 Related Work

### 2.1 Learning Object Placements.

There have been several attempts to solve the task of object placement with deep learning. Tan et al [29] proposed a branching CNN to jointly predict the location and size for inserting person into a scene. Lin et al [20] proposed Spatial Transformer Generative Adversarial Networks (ST-GAN) that iteratively warps a foreground instance into a background scene with a spatial transformer network via adversarial training against geometric and natural image manifolds. Similarly to [20], Tripathi et al [31] proposed to composite synthetic images with STN [14] by discriminating them from the natural image datasets. Azadi et al [1] proposed a self-consistent composition-by-decomposition network named Compositional GAN to composite a pair of objects. The insight is that the composite images should not only look realistic in appearance but also be decomposable back into individual objects, which provides the self-consistent supervisory signal for training the composition network. Li et al [19] focused on predicting a distribution of locations and poses of humans in 3D indoor environments using Variational Auto-Encoders [16].

The work closest to ours is Lee et al [18], where they proposed a two-step model that predicts a distribution of possible locations where a specific class of objects (person/car) could be placed and how the shape of the class of objects could look like using semantic maps. In contrast with [18], we learn object placements using images of objects and backgrounds as input without compressing them to abstract category names. Using image appearances as input is much harder due to large feature dimensionality, but it allows us to create more contextually natural scenes compared to using GAN generated objects.

### 2.2 Data Augmentation for Object Detection

There have been many efforts to improve performance of object detection or instance segmentation through data augmentation. The most straightforward method to accomplish this is through geometric transformations of the images [11][8][22][28] such as scale changes, horizontal flips, cropping, and rotations. By varying the levels of context around objects, the orientation, and the size of objects, their aim is to augment the data distribution that better matches the natural distribution of objects. Another method includes adjusting the signal-to-noise ratio to model the uncertainty in object boundaries and other possible sampling noises [8] by distorting the color information.

It has been demonstrated that context plays a key role in vision recognition systems [26][30]. Having contextually related objects in a scene has more

of an multiplicative effect than an additive one. That is, a scene composed of contextually sound objects is more than the sum of the constituent parts. Both [26][30] validate that having contextually related objects provides more evidence for recognition than beyond just the local evidence of the object instance itself.

Instead of operating on the original data, one way to generate new images is to cut-and-paste object instances onto an image [5][7][6][10][32]. This has been shown to be effective for both object detection and instance segmentation.

The context-based cut-and-paste method most related to our work is [5], in that placement is learned based on context. But [5] does not condition the placement of the object on both the context and the appearance of the instance itself like ours. Instead the locations are classified on which class is most likely to be present in each location given the context. The method used is unable to distinguish if specific object instances of the same semantic class actually belong in the scene given the context.

Another closely related work is [7], which perturbs the object locations in the original image to augment object-context relationships. In contrast with this work [7], we can sample directly from the joint distribution of three disjoint variables: object appearance, scene appearance, and stochastic variations of object-scene interaction, without being forced in the original background context. This allows us to generate a far greater diversity of scene compositions.

## 3 Methods

Our work aims to learn the placement of foreground objects into background scenes without heavy human labeling. To do so, we first propose a novel data acquisition technique to generate training data for free. Then, we propose a generative model PlaceNet to predict a set of diverse and plausible locations and scales to insert foreground object into background scene. With the learned PlaceNet, we further propose a data augmentation pipeline to shuffle foreground objects into many different background scenes to composite new training data in order to boost object detection and instance segmentation performance.

### 3.1 Data Acquisition by Inpainting

What kind of data do we need in order to learn common sense object placements? Intuitively, our training set needs to contain paired examples of a foreground object, a cleaned background scene without objects, and labeled plausible locations to place the object. While such labeled data would be extremely difficult and expensive to obtain, we propose a novel data acquisition system. Our system leverages existing instance segmentation dataset and a self-supervised image inpainting network to generate the necessary training data for learning object placement.

Our insight is that we can generate such training data by removing objects from the background scenes. With an instance segmentation mask, we first cut

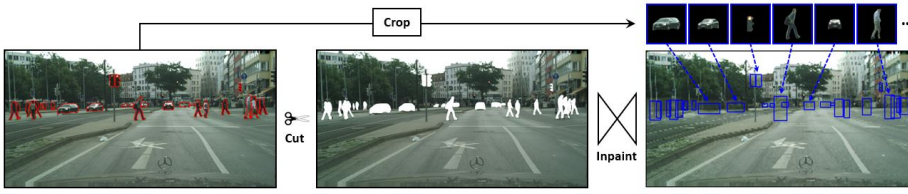


Fig. 2: In our data acquisition pipeline, we first cut out the object region with the instance segmentation mask, and save the original bounding boxes as the ground truth plausible placement locations and scales. In the meantime, we crop out segmented objects corresponding to the bounding boxes. Finally, we use inpainting network to fill the holes of the occluded region and generate the clean background.

out the object regions and then fill in the holes with an image inpainting network. After that, we simultaneously obtain a clean background scene without objects in it and the corresponding ground truth plausible placement locations and scales for placing these objects into the scene. The overall process is described in Figure (2). The instance segmentation can be obtained from labeled data or a pretrained Mask R-CNN network [11]. The inpainting network [33] is trained by randomly cropping out regions in the street scene images. After the training, the inpainting network learns a prior to fill the holes with background information even if the holes were previously occupied by some objects, which has been studied in [2]. Overall, our proposed data acquisition technique provides a way to generate large-scale training data for learning object placement without any human labeling.

### 3.2 Learning Object Placements

Objects can have a multitude of possible placements in a given scene. For example, a person could stand on the left or right side of the street, walk across the street, or stand besides a car. To model such diverse and dense object placements is challenging, since the observation of real-world object placements could be sparse. In order to tackle this problem, we design our PlaceNet to achieve two major properties. First, our model is able to share information across sparse observations of foreground and background affordance in order to accumulate knowledge for dense placement predictions. Second, our model has the ability to actively explore diverse possible solutions for object placements.

To share information across sparse observations, our insight is that objects with similar poses and background with similar layouts could share the observed object placement with each other. Therefore, we encode foreground objects and background scenes into two compact feature vectors, where the foreground feature encodes the object semantics and pose and the background feature encodes background layout. We demonstrate that the learned features can indeed encode such information through image retrieval and feature visualization in section 4.7. With the foreground and background features, we further concatenate them with

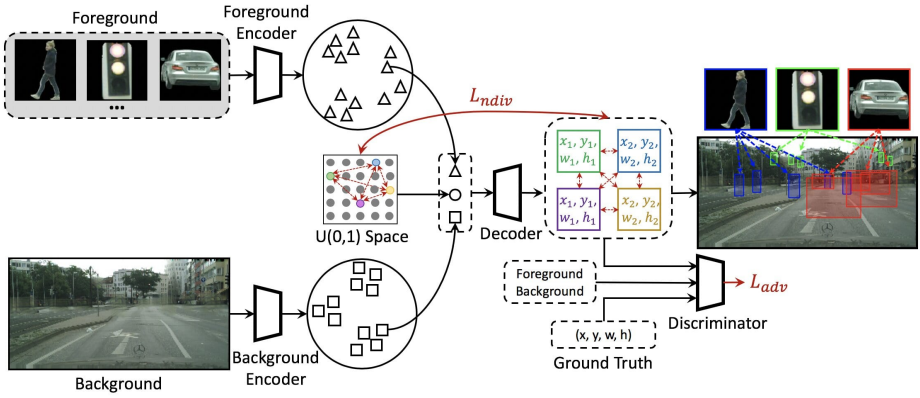


Fig. 3: This is an overview of our PlaceNet. We first encode foreground and backgroud into compact feature vectors, combine them with a random variable sampled from a $U(1,0)$ uniform space, and finally decode to the predicted object placement. The plausibility of predicted placement is checked by a discriminator conditioned on the foreground and the background. The diversity of object placement is achieved by preserving the pairwise distance between predicted placements and the corresponding random variables. The green, blue, yellow, purple circles and boxes denote the sampled random variables and the corresponding predicted placements respectively, and the red dashed double-arrow lines denote the pairwise distance.

a random variable sampled from $U(1,0)$ uniform distribution, and finally decode to a predicted object placement. In this work, we parameterize object placement as normalized horizontal and vertical locations and scales in the range of $0 \sim 1$.

To achieve active exploration of object placements, our insight is that we can enforce the sampled random variables to generate unique and diverse placement solutions. This is achieved by preserving the pairwise distance of the predicted placements with respect to the pairwise distance of the corresponding random variables in the sampling space [21]. To be more specific, we define the diversity loss as follows in Equation (1).

$$\mathcal{L}_{ndiv}(y,z) = \frac{1}{N^2-N}\sum_{i=1}^{N}\sum_{i\neq j}^{N} max(0, \alpha D_{ij}^{z} - D_{ij}^{y}) \tag{1}$$

$$D_{ij}^{z} = \frac{d_z(z_i,z_j)}{\sum_j d_z(z_i,z_j)} \quad , \quad D_{ij}^{y} = \frac{d_y(y_i,y_j)}{\sum_j d_y(y_i,y_j)} \tag{2}$$

where $z$ denotes the random variable, $y$ denotes the predicted placements, $N$ is the number of sampled random variables, $i,j$ indicate the sample indices, and $\alpha$ is a relaxation hyperparameter in the hinge loss. In equation (2), $D_{ij}^{z}$, $D_{ij}^{y} \in \mathbb{R}^{N\times N}$ are the normalized pairwise distance matrices. The distance metric $d(\cdot,\cdot)$ for random variable $z$ and placement $y$ is simply the Euclidean distance, which is defined as follows.

$$d_z(z_i, z_j) = ||z_i - z_j|| \quad , \quad d_y(y_i, y_j) = ||y_i - y_j|| \tag{3}$$

In our implementation, we sample four random variables ($N = 4$) at each iteration, and optimize the network to preserve the pairwise distance between the four predicted placements with respect to the four latent variables in the uniform space. With such learning objective, our model is able to produce diverse placement solutions for each pair of foreground and background inputs.

While the diversity loss $L_{ndiv}$ encourages the network to sample diverse placements, we use a conditional GAN loss [24] to check whether the predicted placements are plausible in the meantime. We train a discriminator that takes foreground, background, and object placement as inputs and computes the probability of whether the predicted placement is realistic conditioned on the foreground and background. This conditional adversarial loss is defined as follows,

$$\mathcal{L}_{adv} = E_{x \sim p_{\text{data}}(x)} \left[ log(D(y|f,b)) \right] + E_{z \sim p(z)} \left[ log(1 - D(G(z|f,b)|f,b)) \right] \tag{4}$$

where $D$ is discriminator, $G$ is generator, $f$ is foreground, $b$ is background, $y$ is ground truth placement, $z$ is the random variable, and $G(z|f,b)$ is the predicted placement. To stabilize training, we apply the spectral normalization [25] to scale down the weight matrices in the discriminator by their largest singular values, which effectively restricts the Lipschitz constant of the network.

### 3.3 Data Augmentation

We randomly select a background to start placing objects, but the starting background could be completely empty and filled in with inpainting or only a few objects removed. This allows us to combine the natural distribution of the object placements with our own generated ones. This essentially can generate more contextually natural and more varied scenes around objects. The overall pipeline is shown in Figure (4).

After selecting the background, we then choose objects that are semantically similar to the ones previously removed from the scene. This is done because there can be multiple reasons why two instances of the same class might not belong in the same scenes. The most obvious reason why an object might not belong is that some instances are occluded. For example there are many "floating heads" in Cityscapes [4] because cars are in front of the person. This is done by selecting top K nearest neighbors from the foreground database for each of the previously existing objects in the scene. We use our pretrained encoder to extract features of foregrounds and use cosine similarity as a distance metric to find top K neighbors. Basically, the K nearest neighbors search finds a plausible subset of foregrounds to add into a specific background.

From there, we randomly select an object from a retrieved foreground subset and feed them into PlaceNet together with a background image one at a time. From the predicted locations and scales, we simply cut and paste to synthesize the new image. The method of cut-and-paste has been demonstrated by previous works [32][5][7] to not be detrimental for detection or instance segmentation despite the visual flaws at the borders.

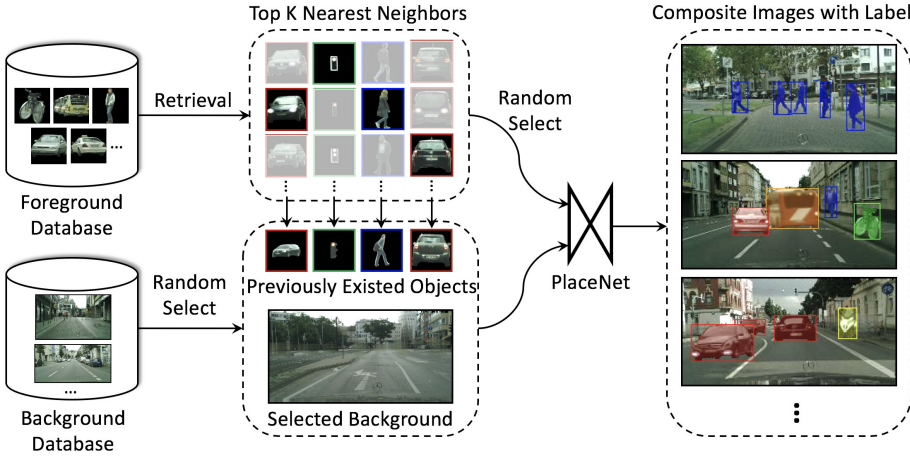


Fig. 4: In our data augmentation pipeline, the foreground database contains masked foreground objects and the background database contains "cleaned" backgrounds with no objects. To make sure the selected foregrounds semantically make sense to be placed into a background, we retrieve top K nearest neighbors of foregrounds with respect to the objects that were previously in the background scene. Then, we randomly select several foregrounds in the retrieved subset of foregrounds and copy-and-paste them into a selected background with predicted locations and scales from our PlaceNet.

Due to the diversity property of the PlaceNet, we are better able to model the probability distribution map of objects in a scene. The modularity of our data composition design allows us to generate any pair of object-context images. These two properties combine to generate novel scenes with contextually related objects that appear sufficiently different yet natural. Another effect of this is that we can decorrelate instances from a specific scene and location using diversity and modularity since objects can be naturally shuffled into different background scenes.

## 4 Experiments

We evaluate the performance of our method through comparison between strong baselines and the state-of-the-arts in three sub-tasks: object placement, data augmentation for object detection and instance segmentation, and feature learning. In the following sub-sections, we first elaborate our baseline methods and implementation details. Then, we dive into the detailed evaluation and discussion for all the experiments.

### 4.1 Baselines

To evaluate the performance of object placements, we proposed three baseline models, which are Random Placement, Regression, cVAE-GAN, and cVAE-

GAN+Div. In addition, we proposed a k-nearest-neighbor Object Swap baseline and an Object Jitter [7] baseline to evaluate the data augmentation.

**Random Placement**: This approach places objects into a scene with randomly sampled location and scale, where the random sampling is bounded by the extreme object location and size in the dataset.

**Regression**: The regression model directly predicts the bounding box location and scale using MSE loss, and does not have stochastic sampling property.

**cVAE-GAN** [17]: The name of cVAE-GAN is conditional Variational Auto-Encoder with Generative Adversarial Network. This model contains a cVAE to stochastically sample outputs, which are followed by a discriminator to check the plausibility of the outputs.

**cVAE-GAN+Div** [17][23]: This model is simply the cVAE-GAN model with an additional diversity regularization loss [23].

**Object Swap**: For each segmented object in the scene, we swap the object with one randomly chosen object from the k-nearest-neighbors in the foreground object database.

**Object Jitter** [7]: This method proposed to inpaint the segmented object and randomly perturb its original locations with a learned probability heatmap to augment the training data variation.

### 4.2 Implementation Details

To introduce our model architectures, we employ the following abbreviation: N = Number of filters, K = Kernel size, S = Stride, P = Padding. The foreground encoder can be represented as N32K4S2P1 - N64K4S2P1 - N32K4S2P1 - N128K4S2P1 - N256K4S2P1 - N512K4S2P1 - N1024K4S1P0 - N128K1S1P0. The output of the foreground encoder is a 128d feature vector. The background encoder has the same layers as the foreground encoder except for the last one, which outputs 256d feature vector to capture background information. The 128d foreground feature and 256d background feature are concatenated and fed into fully connected (fc) layers (386 - 256 - 64 - 4) and outputs the bounding box. All of the layers are followed by a batch normalization [13] and LeakyReLu, except for the last layer, which uses a sigmoid function to normalize the outputs. In the discriminator, we use the same design of foreground and background encoders without sharing weights to extract foreground and background features. Then the two feature vectors are concatenated with the predicted a 4d object placement, and are fed into four fc layers (386 - 256 - 64 - 1) to compute the whether the object placement is realistic conditioned on the foreground and background. All layers except for the last one in discriminator are followed by spectral normalization [25] instead of batch normalization [13] to stabilize training. We use Adam optimizer [15] with learning rate of 2e-4, beta 1 of 0.5, and beta 2 of 0.999, and use batch size of 32. The maximum iteration is set to be 200K.

For our baselines regression, cVAE-GAN and cVAE-GAN+Div, we use the same network architectures for the encoder, decoder, and discriminator modules, and different loss functions. For example, the regression baseline uses MSE loss only. The cVAE-GAN baseline uses KL-Divergence, MSE and adversarial

loss with weights of 0.01, 1, 1 respectively. On top of cVAE-GAN, the cVAE-GAN+Div baseline uses an additional diversity loss [23] with weight of 1 that maximize the ratio of the distance between sampled outputs with respect to the corresponding latent codes.

### 4.3 Object Placements

We evaluate object placement in two criterion: plausibility and diversity. While there is generally a trade-off between plausibility and diversity in generative models [34][23][21], we emphasize that our model aims to produce diverse results without sacrificing the plausibility in the meantime.

For the placement plausibility, we conduct user study that asks user whether the sampled bounding boxes are reasonable for a pair of foreground and background. The final result is averaged across 200 testing examples from ten subjects. In addition, we quantify placement plausibility by computing the Frechet Inception Distance (FID) [12] between composite and real images. Lower FID indicates that the composite distributions are more similar to the real distribution, and that object placements are more realistic and plausible. On the other hand, we compute the diversity of placement by calculating the pairwise Euclidean distance between pairs of sampled object bounding boxes. Overall, the results indicate that our model can generate much more diverse composite locations with even better plausibility scores, as shown in Table (1) and Figure (5).

| Models | Plausibility (User Study) ↑ | Plausibility (FID) ↓ | Diversity ↑ |
|---|---|---|---|
| Random | 22.4% | 70.36 | 0 |
| Regression | 73.1% | 57.86 | 0 |
| cVAE-GAN[17] | 75.9% | 52.13 | 0.0219 |
| cVAE-GAN+Div[17][23] | 74.5% | 53.54 | 0.0335 |
| PlaceNet (Ours) | **76.4%** | **48.15** | **0.0392** |

Table 1: Quantitative evaluation of object placements. Our method achieves the highest diversity as well as the best plausibility.

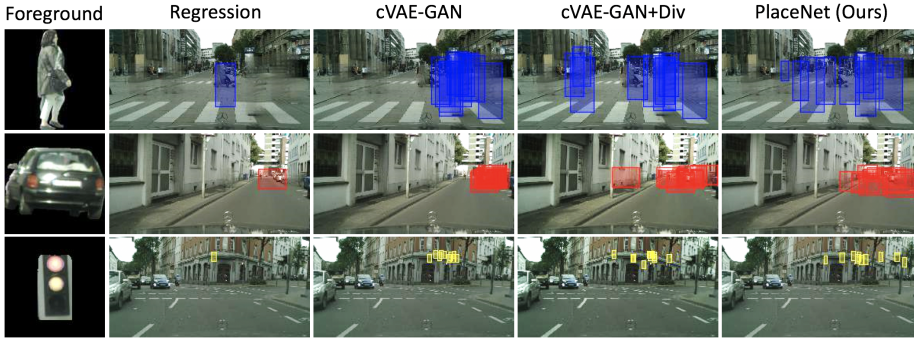


Fig. 5: This is a qualitative comparison of object placement predictions between two baseline models and our PlaceNet. Our method can generate the most diverse object placements in the comparison.

### 4.4 Overfitting Inpainting Artifacts?

We study whether the our PlaceNet will overfit to the inpainting artifacts. Theoretically, since many different objects are inpainted at the same time in an image, the network can not use the artifacts as cues to find reasonable placements for a specific class of objects. For example, artifacts of inpainted cars do not provide cues for placements of traffic lights, and vice versa. Empirically, we evaluate how many generated boxes are covering the original object locations. With a Intersection-over-Union (IoU) threshold of 0.5, we observe that only 37.62% of generated boxes are covering the original locations by sampling across 1,000 examples. In Figure (6), we show four examples that the predicted locations (purple boxes) are not covering the original inpainting object locations (white boxes).

Fig. 6: Inpainting locations (white boxes) and predicted locations (purple boxes). The predicted locations do not covering the inpainting locations most of the time, this indicates that our PlaceNet does not overfit to the inpainting artifacts.

### 4.5 Data Augmentation for Object Detection

To evaluate the data augmentation performance for object detection, we use the same detector YOLOv3 [27] on all augmented datasets and use mean average precision (mAP) as the evaluation metric. The baseline is the model trained with the original dataset only. Compared to this baseline, our method can boost the object detection for all the classes except for car, as shown in the last row of Table (2). This comparison shows that our data augmentation can boost the rare and hard class detection by an obvious margin, such as rider and truck, by generating more rare object-and-context scenes to balance out the original data bias.

Compared to the other methods, our method can achieve the best overall mAP performance boosting on both Cityscape [4] and KITTI [9] dataset. From the random placement baseline, we can see that introducing any wrong contextual relationship could harm the data augmentation and perform even worse than the baseline. By looking at the comparison between cVAE-GAN/cVAE-GAN+Div between our method, we can see that by generating more diverse composite scenes, as shown in Table (1), our method can further boost the data augmentation performance. From the visual comparison in Figure (7), we can

see that our method is able to identify bicycle, person, and cars with better precision and recall.

| Object Detection | Cityscape[4] | | | | | | | | | KITTI[9] |
|---|---|---|---|---|---|---|---|---|---|---|
| | Person | Rider | Car | Truck | Bus | Train | Motorcycle | Bicycle | Overall ↑ | Overall ↑ |
| Baseline | 0.412 | 0.238 | 0.754 | 0.154 | 0.153 | 0.105 | 0.080 | 0.224 | 0.265 | 0.359 |
| Random | 0.454 | 0.278 | 0.738 | 0.104 | 0.135 | 0.099 | 0.058 | 0.218 | 0.260 | 0.203 |
| cVAE-GAN[17] | 0.441 | 0.323 | 0.745 | 0.154 | 0.203 | 0.105 | 0.104 | 0.223 | 0.287 | 0.274 |
| cVAE-GAN+Div[17][23] | 0.462 | 0.293 | 0.753 | 0.214 | 0.170 | 0.145 | 0.059 | 0.248 | 0.293 | 0.322 |
| Object Swap | 0.437 | 0.303 | 0.757 | 0.162 | 0.160 | 0.123 | 0.082 | 0.244 | 0.283 | 0.275 |
| Object Jitter[7] | 0.441 | 0.323 | 0.744 | 0.154 | 0.202 | 0.105 | 0.104 | 0.223 | 0.278 | 0.354 |
| Object Placement (Ours) | 0.448 | 0.381 | 0.749 | 0.200 | 0.179 | 0.140 | 0.088 | 0.227 | **0.302** | **0.371** |
| Improvement over Baseline | 0.036 | 0.143 | -0.005 | 0.046 | 0.026 | 0.035 | 0.008 | 0.003 | 0.037 | 0.012 |

Table 2: Object detection on Cityscape [4] and KITTI [9]. We run YOLOv3 [27] detector on all the data augmentation methods, and evaluate the results using mean average precision (mAP).

Fig. 7: A visual comparison that shows the results from the baseline (top row) and the results from our approach (bottom row) using YOLOv3 [27].

### 4.6 Data Augmentation for Instance Segmentation

Similarly to object detection, we use the instance segmentation algorithm Mask R-CNN [11] and train it on all the augmented datasets with mAP as the evaluation metric. The quantitative evaluation is shown in Table (3). We show that our method can boost individual class mAP for rider, car, bus, train and motorcycle compared to the baseline. Our method can also achieve the best overall mAP compared to all other methods in both Cityscape [4] and KITTI [9] datasets. Overall, we can see similar performance trend as we have seen in object detection. From the quantitative results of both tasks, we can conclude that generating rare object-and-context scenes could alleviate dataset bias, which boosts the recognition performance on the rare classes. In addition, the more diverse the composite scenes could boost more recognition performance.

From the visual comparison in Figure (8), in the first image, there is a small car clearly ahead of the ego vehicle yet the baseline fails to capture it. In the second picture we detect all the bikes on the bike rack. For the third picture, the baseline has a difficult time in crowded scenes, and it misses multiple people. The motorcycle in the last image is completely missed probably due to its low class appearance. Overall, we can detect more highly occluded and small instances where context is more important for identifying them.

| Instance Segmentation | Cityscape[4] | | | | | | | | | KITTI[9] |
|---|---|---|---|---|---|---|---|---|---|---|
| | Person | Rider | Car | Truck | Bus | Train | Motorcycle | Bicycle | Overall ↑ | Overall ↑ |
| Baseline | 0.202 | 0.069 | 0.620 | 0.495 | 0.493 | 0.156 | 0.133 | 0.118 | 0.286 | 0.235 |
| Random | 0.210 | 0.097 | 0.619 | 0.449 | 0.471 | 0.143 | 0.128 | 0.118 | 0.279 | 0.228 |
| cVAE-GAN[17] | 0.210 | 0.093 | 0.616 | 0.477 | 0.460 | 0.162 | 0.112 | 0.113 | 0.281 | 0.243 |
| cVAE-GAN+Div[17][23] | 0.213 | 0.090 | 0.620 | 0.496 | 0.478 | 0.187 | 0.133 | 0.104 | 0.291 | 0.247 |
| Object Swap | 0.221 | 0.087 | 0.621 | 0.481 | 0.481 | 0.155 | 0.129 | 0.117 | 0.287 | 0.254 |
| Object Jitter[7] | 0.202 | 0.164 | 0.627 | 0.465 | 0.479 | 0.196 | 0.143 | 0.121 | 0.300 | 0.281 |
| Object Placement (Ours) | 0.198 | 0.080 | 0.621 | 0.487 | 0.512 | 0.264 | 0.143 | 0.109 | **0.302** | **0.307** |
| Improve over Baseline | -0.004 | 0.011 | 0.001 | -0.008 | 0.019 | 0.108 | 0.010 | -0.009 | 0.016 | 0.072 |

Table 3: Instance segmentation on Cityscape [4] and KITTI [9]. We run Mask R-CNN [11] detector on all the data augmentation methods, and evaluate the results using mean average precision (mAP).

Fig. 8: A visual comparison that shows the results from the baseline (top row) and the results from our approach (bottom row) using Mask R-CNN [11].

### 4.7 Feature Representation Learning

A key property of our PlaceNet is to share information across sparse observations such that we can learn a dense distribution of diverse object placements. This property is achieved in that our foreground and background encoders are able to learn feature representations that can cluster foreground and background based on their semantics and functionality. By clustering objects and scenes in the latent space, our the network can then share the object-and-context relationships from the sparse observations of objects/scenes pairs.

We run the k-nearest-neighbor image retrieval on the foreground and background images using the learned encoders. As shown in Figure (9), foreground features can cluster object pose regardless of appearance and the background features can cluster background scenes with the similar scene layouts. We further visualize the feature activation of background encoders, and find that the background features implicitly segments the street and non-street regions, which encode the street scene layout, as shown in Figure (10).

In addition to image retrieval and feature visualization, we test out how well the pretrained encoder can be used for foreground image classification. In this experiment, we collect 10K foreground images for training and 1K foreground images for testing, and we aim to classify eight classes of semantic objects in Cityscape [4]. We set up three experiment trials, where we use 1K, 5K ,and 10K

| Number of Training Images | Classifier Trained from Scratch | PlaceNet foreground encoder |
|---|---|---|
| 1,000 | 34.3% | 46.5% |
| 5,000 | 52.5% | 74.8% |
| 10,000 | 67.7% | 86.3% |

Table 4: Comparison between model trained from scratch and model fine-tuned on the foreground encoder on image classification. The numbers in the $2^{nd}$ and $3^{rd}$ columns are the classification accuracy on 1K testing images when using different amount of training data.

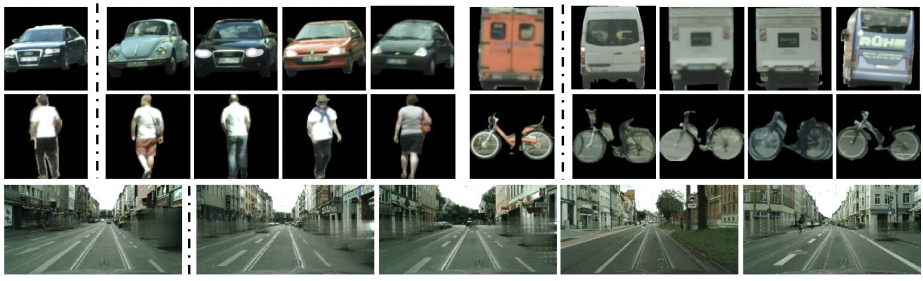

Fig. 9: Foreground and background image retrieval. The foreground encoder can retrieve the objects based on the semantics and pose regardless of color. The background encoder can retrieve the scene based on the street layouts.

Fig. 10: Visualization of feature activation from the background encoder. This shows that the features fire on street segments and non-street segments, which encode the street layout.

training images in each of the three trials. We compare the model trained from scratch and the model that fine-tunes on the learned foreground encoder from PlaceNet. As shown in Table (4), the PlaceNet encoder can consistently outperform the model trained from scratch. This experiment, once again, shows that our PlaceNet can learn meaningful and discriminative feature representation.

## 5 Conclusion

We formulated the self-learning task of object placement. We first proposed a novel data generation technique that can generate large-scale training data for 'free'. Then, we proposed PlaceNet that can learn the distribution of diverse and plausible locations to place a given object into a background. We show that our object placement provides two useful side-effects. First, our learned PlaceNet can be used to shuffle segmented objects into different background scenes to enrich object-context variations for boosting object detection and segmentation. Second, we show that our self-learning PlaceNet can learn meaningful feature representations for object/scene retrieval and classification. Extensive experiments have been conducted to demonstrate the effectiveness of our method compared to the strong baselines and the state-of-the-arts.

## References


1. Azadi, S., Pathak, D., Ebrahimi, S., Darrell, T.: Compositional gan: Learning image-conditional binary composition. arXiv preprint arXiv:1807.07560 (2019) 3
2. Bau, D., Zhu, J.Y., Wulff, J., Peebles, W., Strobelt, H., Zhou, B., Torralba, A.: Seeing what a gan cannot generate. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 4502–4511 (2019) 5
3. Carr, M.F., Jadhav, S.P., Frank, L.M.: Hippocampal replay in the awake state: a potential substrate for memory consolidation and retrieval. Nature neuroscience **14**(2), 147 (2011) 1
4. Cordts, M., Omran, M., Ramos, S., Rehfeld, T., Enzweiler, M., Benenson, R., Franke, U., Roth, S., Schiele, B.: The cityscapes dataset for semantic urban scene understanding. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 3213–3223 (2016) 1, 7, 11, 12, 13
5. Dvornik, N., Mairal, J., Schmid, C.: Modeling visual context is key to augmenting object detection datasets. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 364–380 (2018) 1, 4, 7
6. Dwibedi, D., Misra, I., Hebert, M.: Cut, paste and learn: Surprisingly easy synthesis for instance detection. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 1301–1310 (2017) 1, 4
7. Fang, H.S., Sun, J., Wang, R., Gou, M., Li, Y.L., Lu, C.: Instaboost: Boosting instance segmentation via probability map guided copy-pasting. arXiv preprint arXiv:1908.07801 (2019) 1, 4, 7, 9, 12, 13
8. Fu, C.Y., Liu, W., Ranga, A., Tyagi, A., Berg, A.C.: Dssd: Deconvolutional single shot detector. arXiv preprint arXiv:1701.06659 (2017) 3
9. Geiger, A., Lenz, P., Stiller, C., Urtasun, R.: Vision meets robotics: The kitti dataset. International Journal of Robotics Research (IJRR) (2013) 1, 11, 12, 13
10. Georgakis, G., Mousavian, A., Berg, A.C., Kosecka, J.: Synthesizing training data for object detection in indoor scenes. arXiv preprint arXiv:1702.07836 (2017) 1, 4
11. He, K., Gkioxari, G., Dollár, P., Girshick, R.: Mask r-cnn. In: Proceedings of the IEEE international conference on computer vision. pp. 2961–2969 (2017) 2, 3, 5, 12, 13
12. Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., Hochreiter, S.: Gans trained by a two time-scale update rule converge to a local nash equilibrium. In: Advances in neural information processing systems. pp. 6626–6637 (2017) 10
13. Ioffe, S., Szegedy, C.: Batch normalization: Accelerating deep network training by reducing internal covariate shift. arXiv preprint arXiv:1502.03167 (2015) 9
14. Jaderberg, M., Simonyan, K., Zisserman, A., et al.: Spatial transformer networks. In: Advances in neural information processing systems. pp. 2017–2025 (2015) 3
15. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014) 9
16. Kingma, D.P., Welling, M.: Auto-encoding variational bayes. arXiv preprint arXiv:1312.6114 (2013) 3
17. Larsen, A.B.L., Sønderby, S.K., Larochelle, H., Winther, O.: Autoencoding beyond pixels using a learned similarity metric. arXiv preprint arXiv:1512.09300 (2015) 9, 10, 12, 13
18. Lee, D., Liu, S., Gu, J., Liu, M.Y., Yang, M.H., Kautz, J.: Context-aware synthesis and placement of object instances. In: Advances in Neural Information Processing Systems. pp. 10393–10403 (2018) 3

19. Li, X., Liu, S., Kim, K., Wang, X., Yang, M.H., Kautz, J.: Putting humans in a scene: Learning affordance in 3d indoor environments. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 12368–12376 (2019) 3
20. Lin, C.H., Yumer, E., Wang, O., Shechtman, E., Lucey, S.: St-gan: Spatial transformer generative adversarial networks for image compositing. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 9455–9464 (2018) 3
21. Liu, S., Zhang, X., Wangni, J., Shi, J.: Normalized diversification. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 10306–10315 (2019) 6, 10
22. Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.Y., Berg, A.C.: Ssd: Single shot multibox detector. In: European conference on computer vision. pp. 21–37. Springer (2016) 3
23. Mao, Q., Lee, H.Y., Tseng, H.Y., Ma, S., Yang, M.H.: Mode seeking generative adversarial networks for diverse image synthesis. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 1429–1437 (2019) 9, 10, 12, 13
24. Mirza, M., Osindero, S.: Conditional generative adversarial nets. arXiv preprint arXiv:1411.1784 (2014) 7
25. Miyato, T., Kataoka, T., Koyama, M., Yoshida, Y.: Spectral normalization for generative adversarial networks. arXiv preprint arXiv:1802.05957 (2018) 7, 9
26. Oliva, A., Torralba, A.: The role of context in object recognition. Trends in cognitive sciences **11**(12), 520–527 (2007) 3, 4
27. Redmon, J., Farhadi, A.: Yolov3: An incremental improvement. arXiv preprint arXiv:1804.02767 (2018) 3, 11, 12
28. Ren, S., He, K., Girshick, R., Sun, J.: Faster r-cnn: Towards real-time object detection with region proposal networks. In: Advances in neural information processing systems. pp. 91–99 (2015) 3
29. Tan, F., Bernier, C., Cohen, B., Ordonez, V., Barnes, C.: Where and who? automatic semantic-aware person composition. In: 2018 IEEE Winter Conference on Applications of Computer Vision (WACV). pp. 1519–1528. IEEE (2018) 3
30. Torralba, A., Murphy, K.P., Freeman, W.T., Rubin, M.A.: Context-based vision system for place and object recognition (2003) 3, 4
31. Tripathi, S., Chandra, S., Agrawal, A., Tyagi, A., Rehg, J.M., Chari, V.: Learning to generate synthetic data via compositing. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 461–470 (2019) 3
32. Wang, H., Wang, Q., Yang, F., Zhang, W., Zuo, W.: Data augmentation for object detection via progressive and selective instance-switching (2019) 1, 4, 7
33. Yu, J., Lin, Z., Yang, J., Shen, X., Lu, X., Huang, T.S.: Generative image inpainting with contextual attention. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 5505–5514 (2018) 2, 5
34. Zhu, J.Y., Zhang, R., Pathak, D., Darrell, T., Efros, A.A., Wang, O., Shechtman, E.: Toward multimodal image-to-image translation. In: Advances in neural information processing systems. pp. 465–476 (2017) 10