SemAug: Semantically Meaningful Image Augmentations for Object Detection Through Language Grounding

Morgan Heisler, Amin Banitalebi-Dehkordi, and Yong Zhang

Huawei Technologies Canada Co., Ltd., Burnaby, Canada
{morgan.lindsay.heisler, amin.banitalebi, yong.zhang3}@huawei.com

Abstract. Data augmentation is an essential technique in improving the generalization of deep neural networks. The majority of existing image-domain augmentations either rely on geometric and structural transformations, or apply different kinds of photometric distortions. In this paper, we propose an effective technique for image augmentation by injecting contextually meaningful knowledge into the scenes. Our method of semantically meaningful image augmentation for object detection via language grounding, SemAug, starts by calculating semantically appropriate new objects that can be placed into relevant locations in the image (the `what` and `where` problems). Then it embeds these objects into their relevant target locations, thereby promoting diversity of object instance distribution. Our method allows for introducing new object instances and categories that may not even exist in the training set. Furthermore, it does not require the additional overhead of training a context network, so it can be easily added to existing architectures. Our comprehensive set of evaluations showed that the proposed method is very effective in improving the generalization, while the overhead is negligible. In particular, for a wide range of model architectures, our method achieved 2-4% and 1-2% mAP improvements for the task of object detection on the Pascal VOC and COCO datasets, respectively. Code is available as supplementary.

Keywords: Semantic Image Augmentation, Language Grounding

1 Introduction

Training a deep neural network (DNN) relies on the availability of representative datasets which contain a sufficient number of labeled examples. Collecting relevant samples and labeling them is a time consuming and costly task. In practice, various techniques are employed to improve the network accuracy given the available training data. Of these techniques, methods of artificially expanding the size of the training dataset are of especial importance. For computer vision tasks, image augmentation is a technique that is used to artificially expand the size of a training dataset by generating modified versions of the training images. Almost all modern vision-DNNs involve some form of image augmentation in

Fig. 1. Examples of our method: originals (left) and semantically augmented (right).

training [31]. The importance of augmentation is even more pronounced for applications where training data is imbalanced (distribution of instances among categories is not uniform), when target categories are rare or uncommon in nature (e.g. detection of security threats), or when adding new object categories to datasets.

Although traditional techniques such as flipping, rotating, cropping, and altering the colour space are helpful, they are generic and all-purpose in nature. When performing visual tasks such as object detection and semantic segmentation, a more object-aware method specifically created for these tasks could improve results. To address this, [15],[12] performed studies of placing objects randomly inside training images, and observed consistently better results for both object detection and semantic segmentation tasks.

Conversely, though randomly placing new object instances has an effect of generating more training samples and therefore reduces over-fitting, it is likely forcing the detector to fixate on the appearance of individual objects thereby becoming invariant to contextual information that humans find useful [11]. Intuitively, methods which preserve such context should further boost performance results. This intuition was confirmed by [11], [10] that showed context-based object placement achieves higher generalization compared to random placement. However, training a context model adds a considerable overhead, making it less practical in real-world applications [14]. In addition, contextual associations in such methods are derived using data in the training dataset and therefore the potential for new associations is limited.

Contextual relationships have been an area of interest in the Natural Language Processing (NLP) world for quite some time [1,8,23,16]. In this domain, words can be represented as real-valued vectors allowing for quantitative latent space analysis. Various language models such as GloVe [29], fasttext [27], BERT [8], etc. have been trained on vast text corpora to encode the intricate relationships between words. In this paper, we present a simple and effective method for injecting contextual information using these semantic word vectors, without the overhead of training additional dataset-dependent context networks. By leveraging the semantic labels, our method can consider the context of a scene

Fig. 2. Various methods of augmentation: From left to right: the original image, traditional augmentations (flip, contrast/brightness adjustment, additive noise), random object placement, and SemAug (our method). A giraffe could reasonably be found in a field with elephants, whereas a traffic light has no contextual basis in this scene.

and augment appropriately as shown in Figure 1 through the injection of contextual knowledge. In brief, word vectors from pre-trained embeddings are used to compute the most similar objects which can then be placed in an image. A comparison to other techniques is shown in Figure 2 where the original image consists of elephants in a field with sky above. Traditional techniques are able to globally modify the image to look different than the original, but do not add any information based on prior knowledge. Neither does the random object placement which placed a traffic light in the scene. The semantically augmented image has added a giraffe, which is contextually relevant, and added it to the scene in an appropriate location. This addition based on prior knowledge aids the network in discerning the relationships between objects in a dataset.

The main contributions of this paper are as follows:

– We present a new method of object-based contextually meaningful image augmentation for object detection. In particular, we propose a solution for the `what` and `where` problems for object instance placement. Moreover, our method allows for the introduction of new object categories into a dataset while still considering context as it is not dataset-dependent.
– Our method considers context without the overhead of training additional context models, allowing for easy adoption to existing models and training pipelines.
– Through a comprehensive set of experiments, we show that our method provides consistent improvements on standard object detection benchmarks. We show our context-based object handling is indeed more meaningful than random placements, while it does not require training additional context models.

2 Related works

Related to our work are image augmentation methods, in particular context-based augmentations. We provide a brief overview in this section.
Traditional augmentations: Include rotate, flip, resize, blur, added noise, color manipulations, and other geometric or photometric transformations. A typical preprocessing pipeline may include a combination of such augmentations.
Combining image augmentations: To address situations where traditional augmentations do not cover, several more advanced methods of mixing augmentations and their respective labels have been proposed in the recent years.

Examples include: RandAugment [7]/AutoAugment [6] (to identify suitable augmentations on each training iteration), AugMix [22] (mixing multiple random augmentations and enforcing a consistency loss), MixUp [37] (mixing training samples), CutOut [9] (cutting out a random bounding box), CutMix [36] (cut a random box from an image and paste to another), DeepAugment [21] (adding perturbation on weights and activations), FenceMask [24] (fence-shape CutOut), FMix [18] (applying binary masks from Fourier space), KeepAugment [17] (CutMix but not applying any augmentations on the pasted box), ClassMix [28] (combining segmentation masks), ComplexMix [5] (advanced version of ClassMix). These augmentations have shown to improve on the traditional augmentations, however as mentioned in the introduction section, some form of object-level augmentation specifically created for the task of object detection rather than image classification may provide a larger performance boost.

Object-level augmentations: The previous methods do not consider any specific object-level augmentations but rather apply some transformations over the whole image, which may not be optimal for tasks such as object detection. Recently, object-aware methods such as Copy-Paste [15] or Cut-Paste-Learn [12] have gained traction (denoted by 'random' in Figure 2). Though these methods do increase the number of object instances in a dataset, no prior contextual knowledge is used to determine whether pasted objects would naturally be found in the scene. This is the major disadvantage because the object-aware method may learn improper associations which would not appear in test images, leading to inevitable accuracy loss in object detection.

Other than the random object pasting, some recent methods propose other approaches of object-level augmentation. InstaBoost [14] proposes to move an object within its neighborhood to create new training examples. Inpainting might be used to fill in the black pixels. PSIS [34] and COCP [35] on the other hand, switch different instances of a same object category within two images. While effective, these methods provide sub-optimal augmentation as the object categories for each image do not change. Additionally, the constraints in place for COCP [35] inhibit the number of synthetic images that can be created, especially for a smaller dataset. Our method is able to add new object categories to images, enabling stronger perturbations in the image domain, as well as add new object categories to the datasets.

Contextual augmentations: To take context into consideration, Context-DA [11,10] and [33] proposed to train a separate model that learns the context. The main disadvantages of using an additional context model are: additional networks require extra training overhead, and are highly dataset-dependent. Our method differs as it does not model the visual context of the images, but rather leverages the availability of pre-trained language embeddings to derive semantic context from images. This allows for the injection of new knowledge not necessarily already in the given dataset, less overhead than training and inferencing an additional context network, and improved flexibility as it can be readily used in any architecture or framework.

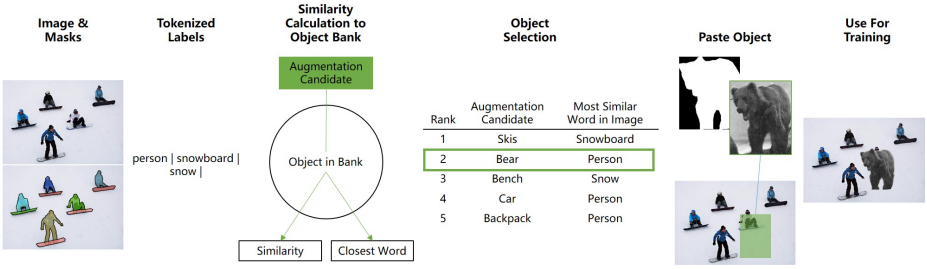

Fig. 3. Illustration of our data augmentation approach. After an image is selected for semantic augmentation, the semantic labels are converted into word vectors. The similarity between these word vectors and the word vectors of the available objects to be pasted are computed. Then one of these objects is chosen from the object bank based on a criteria such as balancing the number of objects in a dataset, or adding more instances of a poor-performing object category. The chosen object is then pasted into the image in the vicinity of the most similar label.

3 Method

In this section, we first provide a formulation of the problem. Then, we describe our method and provide insights on different aspects of our approach.

3.1 Problem statement

Let $\mathbf{I} \in \mathbb{R}^{W \times H}$ denote a training image from the train set with width W and height H (for brevity we drop the channels dimension). The goal of SemAug is to generate a new training sample $\tilde{\mathbf{I}}$ by inserting one or more contextually relevant objects from an object bank Ω using semantic knowledge, π. This can be expressed by:

$$\tilde{\mathbf{I}} = f_{\pi}(\mathbf{I}, \Omega). \tag{1}$$

In this section, we present a method of language grounding as a way of extracting and matching high level semantic context π. The augmented training sample set $\{\tilde{\mathbf{I}}\}$ is then used to train the model with its original loss function. Through this injection of semantic knowledge, we strengthen the network's ability to predict objects given context.

3.2 Semantic augmentation

An overview of SemAug is illustrated in Figure 3 and detailed steps are summarized in Algorithm 1. In our method, we first create an object bank Ω that contains multiple instances of various objects that can be inserted into host images. The object bank can be created either from external sources such as the web, or can be created based upon an existing dataset. Due to its convenience, we opt in for the second option in this work.

Algorithm 1 The proposed semantic augmentation method

Inputs: An image dataset $\mathcal{D}$
Output: Semantically augmented images $\tilde{\mathcal{D}}$

procedure SemAug($\mathcal{D}$)
 # Object Bank Creation
  for each $\mathbf{I}$ in $\mathcal{D}$ **do**:
    $\mathbf{M} \leftarrow$ `GetMask`($\mathbf{I}$)
    for each object k in $\mathbf{I}$ **do**:
      $\mathbf{I}_c^k, \mathbf{M}_c^k = \texttt{crop}(\mathbf{I}^k, \mathbf{M}^k | bbox^k)$
      $\mathbf{L}_e = \texttt{GetLanguageEmbedding}(\mathbf{L}_w(k))$
      $\Omega \leftarrow \mathbf{I}_c^k, \mathbf{M}_c^k, \mathbf{L}_w, \mathbf{L}_e$
  $\mathbf{D} : \mathbf{L}_w \rightarrow \mathbf{L}_e$
 # Image Augmentation
  for each $\mathbf{I}$ in $\mathcal{D}$ **do**:
    $a^*, b^* = \texttt{FindBestMatch}(\mathbf{I}, \Omega)$ from (5)
    $\mathbf{I}^*, \mathbf{M}^* = \texttt{PadZeros}(\mathbf{I}, a^*, b^*)$
    $\tilde{\mathcal{D}} \leftarrow \tilde{\mathbf{I}} = \mathbf{I} \odot (1 - \mathbf{M}^*) + \mathbf{I}^*$

Once the object bank is created, we explore the language representations associated with the objects in the bank and analyze them with respect to the objects appearing in each training image. By matching the high level semantics through the lens of language embeddings we identify `what` and `where` to insert from the bank to a host image. Details are explained in this section.

At the end, we can apply any other kind of augmentation such as the traditional image transformations to the pipeline before passing the dataset to the training engine.

Object bank creation To create the object bank, we first start by generating an approximate segmentation mask $\mathbf{M}$ for each image $\mathbf{I}$ in the dataset. These masks can be generated by leveraging a static side model such as a DeepLab [4] model (later we study the impact of mask quality and observe that even rough masks are enough). For the k^{th} object in the image, its mask can be denoted as $\mathbf{M}^k \in \{0,1\}^{W \times H}$. The mask associates a binary value where the k^{th} object appears in the image, such that $\mathbf{M}_{x,y}^k = 1$ if the pixel at (x, y) belongs to the k^{th} object. The object's masked image can be denoted as $\mathbf{I}^k \in \mathbb{R}^{W \times H}$. Before placing the object's image and mask in the object bank, they are first cropped according to the object's bounding box to reduce their storage space:

$$\mathbf{I}_c^k, \mathbf{M}_c^k = \texttt{crop}(\mathbf{I}^k, \mathbf{M}^k | bbox^k). \tag{2}$$

This process is done once, before training, and for all images in the training dataset. The last step of the object bank creation is to create a dictionary, $\mathbf{D}$, of all the words (or "tokens") in semantic labels, $\mathbf{L}_w$, and their corresponding word embeddings, $\mathbf{L}_e$, such that $\mathbf{D} : \mathbf{L}_w \rightarrow \mathbf{L}_e$. To this end, we leverage an existing language model to extract the embedding descriptions of the semantics.

Matching semantics through word embeddings Once we obtain the language representations of objects, we perform a similarity analysis to identify a target object from the bank (`what`) and where to place it in the host image (`where`). We use the cosine similarity metric to measure the embedding similarity, however other metrics such as a Euclidean distance can be used too. To

Fig. 4. Our method can augment different instances from the same object category. Top row: Different instances from the category airplane are inserted. Bottom row: Different instances from the category kite are inserted.

this end, let a and b denote two word vectors we wish to compare. The cosine similarity is defined as:

$$f_{sim}(a,b) = \frac{a \cdot b}{\|a\| \cdot \|b\|} = \frac{\sum_{i=1}^{d} a_i b_i}{\sqrt{\sum_{i=1}^{d} a_i^2}\sqrt{\sum_{i=1}^{d} b_i^2}}, \tag{3}$$

where d is the word embedding dimension. Supplementary materials [2] contain ablations on the choice of the embedding dimension.
A simple strategy for object selection is to choose the object pair with the highest similarity:

$$a^*, b^* = \underset{a \in \{\mathbf{L}_e^{\mathbf{I}}\}, b \in \{\mathbf{L}_e^{bank}\}}{\text{argmax}} f_{sim}(a,b), \tag{4}$$

where $\{\mathbf{L}_e^{\mathbf{I}}\}$ and $\{\mathbf{L}_e^{bank}\}$ denote all possible embedding choices within the host image and the bank, respectively, and a random instance from the b^* object category will be inserted in the host image at the a^* location. While this strategy intuitively might make sense, it has a down-side that during different epochs, a same object category will be selected every time. To address this issue, we choose from the top N most similar embeddings, the object category with the least number of appearances so far in the current epoch. Note that the number of instance appearances is constantly being updated due to object injection and batch-wise training. Therefore, we are dynamically promoting for a better balancing of the training examples, while also choosing categories with high semantic similarity:

$$a^*, b^* = \underset{b \in \{top\ N\ sim\}}{\text{argmin}} \; count\left(\underset{a \in \{\mathbf{L}_e^{\mathbf{I}}\}, b \in \{\mathbf{L}_e^{bank}\}}{\text{arg-top N}} f_{sim}(a,b) \right). \tag{5}$$

Image augmentation Following the semantic similarity matching strategy of (5), we obtain an object category b^* and a target host object a^*, i.e., `what` and `where`. In this section, we explain the actual image augmentation procedure using the selected pairs. To this end, first, we randomly select an object-image

Fig. 5. Our method can augment instances of different categories. Top row: An instance from the categories truck, bus and motorcycle are inserted. Bottom row: An instance from the categories sheep, and bird are inserted. Note that objects are inserted in logical locations: vehicles on roads, birds in trees, sheep on grass.

instance of type b^* from the bank (Note that different instances can be selected at different epochs, thereby presenting diversified object instances to the training algorithm). Then we scale it randomly by a factor between 5-40% of the image $\mathbf{I}$ width. The object image is scaled using linear interpolation and the mask is scaled using nearest neighbour interpolation. To ensure the pasted objects are not too small or too large, we repeat the random scaling until the resized object's area falls within some bounds (A_{min}, A_{max}). Next, the center coordinates of the incoming object (x_b, y_b) is selected at a random vicinity of the corners of the most similar object in the image, as follows:

$$x_b = x_a \pm \frac{w_a}{2} \pm \epsilon_a, \qquad y_b = y_a \pm \frac{h_a}{2} \pm \epsilon_b, \tag{6}$$

where (x_a, y_a) is the center coordinate of the host object, (w_a, h_a) are its bounding box width and height, and ϵ_a and ϵ_b are small random values to add extra randomness in the placement. If this results in occlusions, the bounding box labels are updated accordingly.

Once the center of the object to be pasted is found, its image and mask are padded with zeros to fit the shape of the training image $\mathbf{I}$. The zero-padded image and mask are denoted as $\mathbf{I}^*$ and $\mathbf{M}^*$ respectively. To compute the final augmented image and mask $(\tilde{\mathbf{I}}, \tilde{\mathbf{M}})$ the followings are used:

$$\tilde{\mathbf{I}} = \mathbf{I} \odot (1 - \mathbf{M}^*) + \mathbf{I}^*, \tag{7}$$

$$\tilde{\mathbf{M}} = \mathbf{M} \odot (1 - \mathbf{M}^*) + \mathbf{M}^*, \tag{8}$$

where $\odot$ denotes the element-wise multiplication. At this point, the semantically augmented image $\tilde{\mathbf{I}}$ is ready to be used for training. Semantic augmentation examples are seen in Figure 4, where the method pasted different instances from the same category and Figure 5 where different categories were pasted into the image. In both figures, the pasted objects are contextually relevant to the scenes.

3.3 Computational complexity

In its simplest form, SemAug uses a dictionary lookup to gather the word embeddings of an image, computes a similarity metric then chooses an object to be pasted based on the similarity values. The complexity of the initial creation of the dictionary is O(len(D)) where len(D) is the number of dictionary items. To get a value from the dictionary is O(1), therefore for each image it is O(obj) where obj is the number of labeled objects in the image. For cosine similarity, the overall computational complexity is O(len(D).Obj.d), where d is the dimension of the embeddings, as the similarity is being calculated for every pair of word embeddings. This negligible overhead is the extra computation that is required to take place for each image during training. For a Mask-RCNN model with ResNet-50 backbone trained on COCO, the additional FLOPs required will be 480,000 (80 objects×20 objects×300 dimension vector). This corresponds to only 0.000107% additional FLOPs. Inference does not incur any extra overhead as it is unchanged.

4 Experiments

This section reviews the results of experiments in support of our method, and provides discussions around them.

4.1 Setup

Architecture: For a fair comparison with existing cut-paste methods, we used Mask R-CNN [19] with ResNet [20] backbone and the publicly available MMDet toolkit [3] on the MS COCO dataset [26]. We also show that SemAug is compatible with Faster-RCNN [30] and RetinaNet [25] using this framework in addition to showing that it improves data efficiency. Otherwise, we employ an Efficientdet-d0 [32] as the backbone for some PASCAL VOC experiments. We ran the experiments on a server equipped with eight NVIDIA V100 GPUs.
Training details: For the experiments in this paper, we choose a default N of 3, for the top 3 most similar embeddings. For (A_{min}, A_{max}) we use the values (300, 90000). Additionally, default image resolutions from MMDet/Efficientdet config files were used [3], [32].
Datasets: We evaluate SemAug on two standard benchmarks: MS COCO [26] and Pascal VOC [13]. The COCO dataset contains 118k training, 5k validation images, and 41k test images over 80 object categories. The Pascal dataset is considerably smaller containing only 20 object categories. Following the standard practice, we use VOC'07+12 training set (16551 examples) for training, and evaluate the models on the VOC'07 test set (4952 images). In contrast to previous object-based approaches such as [35] and [14] which relied on accurate ground-truth segmentation masks, in our method these masks were generated with an off-the-shelf DeepLab-v2 [4] model when needed (See Section 4.2 for details).

For language grounding, we used the word embeddings from Glove [29] trained over a 2014 Wikipedia dump + Gigaword 5 [29] with a dimension of 300.

4.2 Results

Comparison to cut-paste methods: In this subsection, we compare with state-of-the-art cut-paste methods (e.g., COCP [35], InstaBoost [14] and Context-DA [11]) using Mask R-CNN based on ResNet101 on object detection and instance segmentation tasks. The results can be seen in Table 1 where our SemAug outperforms ContextDA [11], InstaBoost [14] and COCP [35] by 2.8%, 2.1% and 1.6% on object detection, respectively ('Vanilla' refers to traditional augmentations used by default in MMdet training pipeline, and is applied for all benchmarks). On the COCO test-dev dataset, SemAug achieves 41.6% mAP, while Vanilla 39.4%, and Instaboost 39.5%. Additionally, our method sees similar performance boosts on the task of instance segmentation. Specifically, our SemAug outperforms ContextDA [11], InstaBoost [14], and COCP [35] by 2.4%, 1.7%, and 1.5%, respectively. An additional comparison to Context-DA is provided in the supplementary materials. Based on these observations, our SemAug method achieves better accuracy than other cut-paste approaches.

	APdet, IOU			APdet, Area			APseg, IOU			APseg, Area		
	0.5:0.95	0.50	0.75	Sma.	Med.	Lar.	0.5:0.95	0.50	0.75	Sma.	Med.	Lar.
Vanilla [19]	39.6	61.4	43.5	23.1	43.8	51.5	36.0	57.9	38.7	19.0	39.7	49.5
Context-DA [11]	39.9	61.4	43.7	23.0	44.2	51.5	36.2	58.2	38.4	19.4	39.8	49.9
InstaBoost [14]	40.6	62.1	44.3	24.4	44.6	53.3	36.8	58.6	39.6	20.4	40.4	50.8
COCP [35]	41.1	62.5	45.0	23.3	44.6	52.4	37.0	58.9	39.4	19.4	40.5	50.7
SemAug	**42.7± 0.13**	**64.5**	**46.9**	**25.6**	**47.3**	**56.1**	**38.5± 0.11**	**61.3**	**41.1**	**21.7**	**42.3**	**53.4**

Table 1. Comparison to other state-of-the-art (SOTA) methods using MMdet and Mask RCNN with a Resnet 101 backbone on COCO val. Context-DA and COCP numbers taken from the COCP paper [35]. The APdet and APseg for SemAug are reported as the mean value and 95% Confidence Intervals based on 5 repeat trails.

Results of incorporating SemAug in labeled datasets and different architectures: Our SemAug method has been shown to work on a variety of state-of-the-art object detection architectures with different capacities as shown in Table 2. This exemplifies how our augmentation strategy considers context without the training and inference overhead of an additional context models allowing for easy adoption into existing models.

Results using smaller dataset sizes: In many real-world applications, it is difficult to collect and label data. Therefore, we evaluated the performance of our method in settings where less labeled data was available. As shown in Figure 6, SemAug was able to provide a boost in performance even in the low data regimes using a fraction of the COCO dataset.

	Detector	Backbone	APdet, IOU			APdet, Area			ARdet, #Det			ARdet, Area		
			0.5:0.95	0.50	0.75	Sma.	Med.	Lar.	1	10	100	Sma.	Med.	Lar.
Vanilla	Faster R-CNN [30]	Resnet-50	36.5	58.4	39.5	21.7	40.2	46.8	30.5	49.3	51.9	32.8	56.2	65.2
SemAug			**38.5**	**60.7**	**41.5**	**23.9**	**42.4**	**49.5**	**31.7**	**50.4**	**53.0**	**34.8**	**57.6**	**66.7**
Vanilla	Faster-RCNN [30]	Resnet-101	38.5	60.3	41.7	22.7	42.9	49.7	31.7	50.6	53.3	34.8	57.7	66.9
SemAug			**40.5**	**62.8**	**44.4**	**26.1**	**45.1**	**52.0**	**32.8**	**52.0**	**54.8**	**37.4**	**59.6**	**68.8**
Vanilla	RetinaNet [25]	Resnet-50	35.3	55.2	37.6	19.4	39.3	46.5	30.4	49.2	52.3	31.9	56.4	66.9
SemAug			**37.4**	**57.7**	**40.3**	**22.3**	**41.4**	**49.5**	**31.7**	**50.4**	**53.6**	**33.5**	**58.3**	**68.6**
Vanilla	RetinaNet [25]	Resnet-101	37.6	57.5	40.2	20.8	42.2	49.9	31.7	50.6	53.8	33.2	58.4	69.7
SemAug			**39.6**	**60.0**	**42.4**	**23.4**	**44.6**	**52.3**	**32.9**	**51.9**	**55.2**	**35.5**	**60.3**	**71.1**
Vanilla	Mask-RCNN [19]	Resnet-50	37.8	59.5	41.0	23.2	41.4	49.4	31.7	50.6	53.3	35.1	57.5	66.8
SemAug			**39.2**	**61.4**	**42.9**	**24.8**	**43.2**	**50.9**	**32.2**	**51.2**	**53.9**	**35.8**	**58.2**	**68.1**
Vanilla	Mask-RCNN [19]	Resnet-101	39.6	61.4	43.5	23.1	43.8	51.5	32.3	51.5	54.2	34.9	58.8	68.5
SemAug			**42.7**	**64.5**	**46.9**	**25.6**	**47.3**	**56.1**	**34.2**	**54.4**	**57.3**	**38.7**	**62.1**	**71.8**

Table 2. Object detection results (%) on the COCO val benchmark with different size backbones and default parameters.

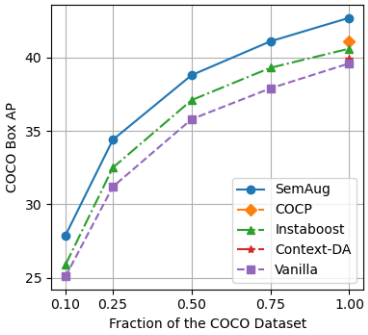

Fig. 6. Data-efficiency on the COCO val benchmark using Mask-RCNN with a Resnet-101 backbone. The results show a consistent increase of $\approx 3\%$ mAP over vanilla in both the low data and high data regimes. Curves (fractional results) are shown for methods for which code was available and could run.

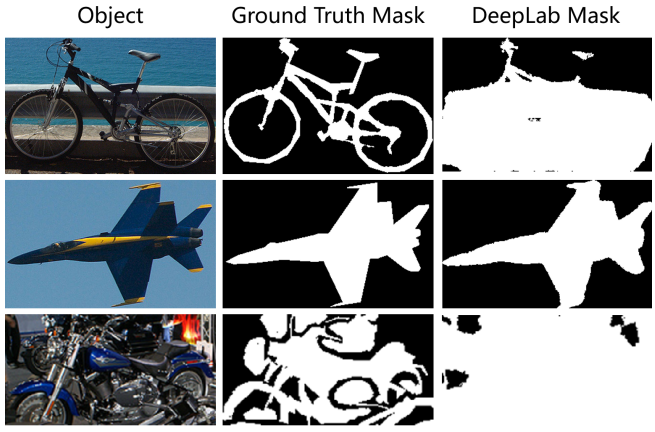

Fig. 7. Mask quality examples. Ground truth masks are much more precise than DeepLab masks.

Ablation of object bank size and mask qualities: As mentioned in Section 3.2, to create an object bank from images without given masks, we can use an off-the-shelf model for convenience such as deeplab. Note the deeplab generated masks are only being used for object bank creation, and therefore the algorithm is not sensitive to their quality. In the case of a bad quality mask (larger or smaller mask) the impact will be similar to either adding an object with more context, or an occluded (partial) object, which may in fact improve the generalization. As shown in Figure 7, the deeplab masks were less precise than the ground truth masks which were only provided for a small subset (VOC-seg, 1464 images) of the VOC dataset. For this experiment we used Efficientdet-d0 with deeplab masks on VOC-seg as well as the whole Pascal VOC training dataset. As shown in Table 3, our method is not sensitive to the quality of the masks in the object bank. An added benefit to using deeplab masks is the ability to supplement the object bank with additional objects from previously unlabeled images. In this regard, we observe that while the deeplab masks were worse quality than the ground truth masks, they provided better performance when additional objects were added to the object bank.

Method	Bank Dataset	Object Mask	mAP
Vanilla	—	—	73.59
SemAug	VOC-seg	DeepLab	77.19
SemAug	VOC-seg	Ground Truth	77.31
SemAug	VOC-all	DeepLab	**77.35**

Table 3. Effect of object bank mask quality on SemAug.

Results of adding new categories: The ability to add new categories to datasets is applicable to important real-world scenarios when target categories are rare or uncommon in nature (e.g. detection of security threats). Due to the inherent constraints of previous works [14], [35], [11], they are unable to add new categories in a knowledgeable manner. To demonstrate SemAug's ability to add new categories, we use Efficientdet-d0 on the Pascal VOC dataset and remove all images of a specific category. Object instances from that category are then pasted into the remaining images during training where appropriate. For this experiment, we chose an N of 5, and only paste objects from the removed category if they are in the top 5 most similar embeddings. We compare the average AP of the other 19 categories before and after the addition of a new category to show that it does not harm the other categories. We do this experiment on the first five categories of VOC, one at a time. Results are shown in Table 4. As observed, SemAug is able to add categories with decent results, while not harming the detection of existing categories in the dataset.

	Aeroplane	Bicycle	Bird	Boat	Bottle
Categories before	79.3	79.5	79.4	80.7	81.1
Categories after	79.6	79.7	79.6	80.6	80.9
New category	64.6	78.2	67.4	39.9	37.9

Table 4. Effect of removing categories from the PASCAL VOC dataset then adding them back using semantic augmentation.Top two rows are the mAP results of all categories except the newly augmented category. Results are recorded as mAP.

Additional comparisons on Pascal VOC: In section 4.2, we compared SemAug with several other SOTA methods on the COCO dataset. Here, we additionally provide a comparison with other augmentation methods on the Pascal VOC object detection task. As in previous papers [36,35], we employ a Faster-RCNN network with a Resnet-50 backbone. The results are given in Table 5. In this table, Random Paste (pasting random objects at random locations) and Co-occurrence (where we paste objects based on how often they appears together in a same image) are two naive object-based augmentation approaches that are included as additional ablation results to our method. As mentioned previously, context is important for the object selection strategy in cut and paste methods. As we can see, methods which do not consider context either degrade performance or marginally improve it; whereas, the three methods that consider context improve performance the most.

Augmentation Method	mAP
Baseline	75.6
Mixup* [37]	73.9 (-1.7)
Cutout* [9]	75.0 (-0.6)
Random Paste	75.9 (+0.3)
CutMix* [36]	76.7 (+1.1)
COCP* [35]	77.4 (+1.8)
Co-occurrence	79.3 (+3.7)
SemAug	**80.7** (+5.1)

Table 5. Comparison to other augmentation methods on the Pascal VOC dataset using Faster-RCNN and a Resnet-50 backbone. * Results taken from [36] and [35].

Ablation of the effect of scaling objects: In this experiment, we compare the use of different scaling ranges with our method. This experiment was conducted using MMDET and the Pascal VOC dataset. Inserting an object can occlude other objects in the scene, and adding an object that is large may remove context from the image. As can be seen in Table 6, it is advantageous to scale the objects so that they are not too small as to be unrecognizable, but also not too big to be occluding other objects.

Table 6. Effect of the scaling range for objects pasted into the scene on the Pascal VOC dataset using Faster-RCNN with a Resnet 50 backbone. The objects are randomly scaled to a percentage of the image into which they are being pasted.

Scaling Range (%)	mAP
No scaling	79.9
5-40	**80.7**
10-30	80.0
15-40	80.4

Ablation on the object similarity metric: In this experiment, we study the impact of object similarity methods discussed in the paper. We employ an Efficientdet-d0 [32] as the backbone, train using the VOC'07+12 training set, and evaluate the models on the VOC'07 test set. As can be seen in the results of Table 7, both euclidean distance and cosine similarity provide comparable results. As cosine similarity provided marginally better results, it was used as default in the paper.

Table 7. Effect of object similarity calculation choice on Pascal VOC.

Object Similarity Method	AP50
Euclidean Distance	77.16
Cosine Similarity	**77.35**

4.3 Limitations

As with any method, there are several limitations to the method presented. Firstly, this method uses pre-exisiting open-source word embeddings. Though this is not a core part of our method and one could choose to train their own word embeddings if necessary. Additionally, the quality of the word embeddings is related to the corpus used for training, therefore care should be taken to ensure meaningful semantic correlations exist before using for augmentation. For example, using a news-based corpora could align 'apple' more with technology than fruit. As several high quality pre-existing open-source word embeddings currently exist, this should not pose a major issue to anyone wishing to use this method. A future works section is discussed in Supplementary [2].

5 Conclusion

This paper proposes an effective technique for image augmentation by injecting contextually meaningful knowledge into training examples. Our object-level augmentation method identifies the most suitable object instances to be pasted into host images, and chooses appropriate target regions. We do that, by analyzing and matching objects and target regions through the lens of high level natural language. Our method results in consistent generalization improvements on various object detection benchmarks.

References

1. Allen, J.: Natural language understanding. Benjamin-Cummings Publishing Co., Inc. (1988) 2
2. Authors: SemAug: Semantically Meaningful Image Augmentations for Object Detection Through Language Grounding (2022), supplied as additional material `5739-supp.pdf` 7, 14
3. Chen, K., Wang, J., Pang, J., Cao, Y., Xiong, Y., Li, X., Sun, S., Feng, W., Liu, Z., Xu, J., Zhang, Z., Cheng, D., Zhu, C., Cheng, T., Zhao, Q., Li, B., Lu, X., Zhu, R., Wu, Y., Dai, J., Wang, J., Shi, J., Ouyang, W., Loy, C.C., Lin, D.: MMDetection: Open mmlab detection toolbox and benchmark. arXiv preprint arXiv:1906.07155 (2019) 9
4. Chen, L.C., Papandreou, G., Kokkinos, I., Murphy, K., Yuille, A.L.: Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. IEEE transactions on pattern analysis and machine intelligence **40**(4), 834–848 (2017) 6, 9
5. Chen, Y., Ouyang, X., Zhu, K., Agam, G.: Mask-based data augmentation for semi-supervised semantic segmentation. arXiv preprint arXiv:2101.10156 (2021) 4
6. Cubuk, E.D., Zoph, B., Mane, D., Vasudevan, V., Le, Q.V.: Autoaugment: Learning augmentation strategies from data. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 113–123 (2019) 4
7. Cubuk, E.D., Zoph, B., Shlens, J., Le, Q.V.: Randaugment: Practical automated data augmentation with a reduced search space. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops. pp. 702–703 (2020) 4
8. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: Bert: Pre-training of deep bidirectional transformers for language understanding. In: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers). pp. 4171–4186 (2019) 2
9. DeVries, T., Taylor, G.W.: Improved regularization of convolutional neural networks with cutout. arXiv preprint arXiv:1708.04552 (2017) 4, 13
10. 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) 2, 4
11. Dvornik, N., Mairal, J., Schmid, C.: On the importance of visual context for data augmentation in scene understanding. IEEE transactions on pattern analysis and machine intelligence (2019) 2, 4, 10, 12
12. 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) 2, 4
13. Everingham, M., Van Gool, L., Williams, C.K., Winn, J., Zisserman, A.: The pascal visual object classes (voc) challenge. International journal of computer vision **88**(2), 303–338 (2010) 9
14. Fang, H.S., Sun, J., Wang, R., Gou, M., Li, Y.L., Lu, C.: Instaboost: Boosting instance segmentation via probability map guided copy-pasting. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 682–691 (2019) 2, 4, 9, 10, 12
15. Ghiasi, G., Cui, Y., Srinivas, A., Qian, R., Lin, T.Y., Cubuk, E.D., Le, Q.V., Zoph, B.: Simple copy-paste is a strong data augmentation method for instance segmentation. arXiv preprint arXiv:2012.07177 (2020) 2, 4

16. Gokhale, T., Banerjee, P., Baral, C., Yang, Y.: MUTANT: A training paradigm for out-of-distribution generalization in visual question answering. In: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP). pp. 878–892. Association for Computational Linguistics, Online (Nov 2020). https://doi.org/10.18653/v1/2020.emnlp-main.63, https://aclanthology.org/2020.emnlp-main.63 2
17. Gong, C., Wang, D., Li, M., Chandra, V., Liu, Q.: Keepaugment: A simple information-preserving data augmentation approach. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 1055–1064 (2021) 4
18. Harris, E., Marcu, A., Painter, M., Niranjan, M., Hare, A.P.B.J.: Fmix: Enhancing mixed sample data augmentation. arXiv preprint arXiv:2002.12047 **2**(3), 4 (2020) 4
19. 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) 9, 10, 11
20. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778 (2016) 9
21. Hendrycks, D., Basart, S., Mu, N., Kadavath, S., Wang, F., Dorundo, E., Desai, R., Zhu, T., Parajuli, S., Guo, M., et al.: The many faces of robustness: A critical analysis of out-of-distribution generalization. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 8340–8349 (2021) 4
22. Hendrycks, D., Mu, N., Cubuk, E.D., Zoph, B., Gilmer, J., Lakshminarayanan, B.: Augmix: A simple data processing method to improve robustness and uncertainty. arXiv preprint arXiv:1912.02781 (2019) 4
23. Hirschberg, J., Manning, C.D.: Advances in natural language processing. Science **349**(6245), 261–266 (2015) 2
24. Li, P., Li, X., Long, X.: Fencemask: A data augmentation approach for pre-extracted image features. arXiv preprint arXiv:2006.07877 (2020) 4
25. Lin, T.Y., Goyal, P., Girshick, R., He, K., Dollár, P.: Focal loss for dense object detection. In: Proceedings of the IEEE international conference on computer vision. pp. 2980–2988 (2017) 9, 11
26. Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft coco: Common objects in context. In: European Conference on Computer Vision (ECCV). pp. 740–755. Springer (2014) 9
27. Mikolov, T., Grave, E., Bojanowski, P., Puhrsch, C., Joulin, A.: Advances in pre-training distributed word representations. In: Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018) (2018) 2
28. Olsson, V., Tranheden, W., Pinto, J., Svensson, L.: Classmix: Segmentation-based data augmentation for semi-supervised learning. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision. pp. 1369–1378 (2021) 4
29. Pennington, J., Socher, R., Manning, C.D.: Glove: Global vectors for word representation. In: Empirical Methods in Natural Language Processing (EMNLP). pp. 1532–1543 (2014), http://www.aclweb.org/anthology/D14-1162 2, 9
30. Ren, S., He, K., Girshick, R., Sun, J.: Faster r-cnn: Towards real-time object detection with region proposal networks. Advances in neural information processing systems **28**, 91–99 (2015) 9, 11
31. Shorten, C., Khoshgoftaar, T.M.: A survey on image data augmentation for deep learning. Journal of Big Data **6**(1), 1–48 (2019) 2

32. Tan, M., Pang, R., Le, Q.V.: Efficientdet: Scalable and efficient object detection. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 10781–10790 (2020) 9, 14
33. Volokitin, A., Susmelj, I., Agustsson, E., Van Gool, L., Timofte, R.: Efficiently detecting plausible locations for object placement using masked convolutions. In: European Conference on Computer Vision (ECCV). pp. 252–266. Springer (2020) 4
34. Wang, H., Wang, Q., Yang, F., Zhang, W., Zuo, W.: Data augmentation for object detection via progressive and selective instance-switching. arXiv preprint arXiv:1906.00358 (2019) 4
35. Wang, H., Wang, Q., Zhang, H., Yang, J., Zuo, W.: Constrained online cut-paste for object detection. IEEE Transactions on Circuits and Systems for Video Technology (2020) 4, 9, 10, 12, 13
36. Yun, S., Han, D., Oh, S.J., Chun, S., Choe, J., Yoo, Y.: Cutmix: Regularization strategy to train strong classifiers with localizable features. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 6023–6032 (2019) 4, 13
37. Zhang, H., Cisse, M., Dauphin, Y.N., Lopez-Paz, D.: mixup: Beyond empirical risk minimization. arXiv preprint arXiv:1710.09412 (2017) 4, 13