Target Driven Instance Detection

Phil Ammirato¹, Cheng-Yang Fu¹, Mykhailo Shvetsk¹, Jana Košecká², Alexander C. Berg¹

Abstract—While state-of-the-art general object detectors are getting better and better, there are not many systems specifically designed to take advantage of the instance detection problem. For many applications, such as household robotics, a system may need to recognize a few very specific instances at a time. Speed can be critical in these applications, as can the need to recognize previously unseen instances. We introduce a Target Driven Instance Detector (TDID), a novel architecture for instance detection. TDID not only improves performance on instances seen during training, with a fast runtime, but is also able to generalize to detect novel instances. We demonstrate superior performance compared to standard object detection models as well as more traditional recognition approaches based on hand-engineered features on modern, challenging datasets.

I. INTRODUCTION

Object detection works! Alas, this is not always true, and the specific version of object detection matters. There have been massive improvements in the accuracy of category-level object detectors based on deep learning [1], [2]. These require many labeled training examples of bounding boxes for each category (e.g. mug) in question, use carefully crafted data augmentation approaches to fully leverage that training data, and can take days or longer to train. This leaves out an important type of object detection problem where the goal is to detect a precise instance of an object category (e.g. my mug instead of a mug). This setting applies to real world tasks including fetch and deliver in household environments, and robotic manipulation in industrial environments, where the objects in question are often specific instances and not general categories. The instance task does not have the large intra-class variation of category-level detection, and sometimes only a small number of training examples for each instance is available.

How can the progress on category-level object detection be harnessed and applied to instance detection, taking advantage of the specificity of instances and overcoming the challenge of small numbers of training examples? One approach is to take a small number of example images and artificially create a large number of detection training examples by artificially composing those examples into scenes [3], [4]. This still treats instance detection as a category detection problem, but expands a small number of clean images of an object instance into enough samples to train current category detectors. Another possible approach reduces the training necessary for new targets by preconditioning a network to be robust to varying views of objects. Recent work by [5] has shown good accuracy with such an approach, and that a deep-learning-based method for learning a classifier from single examples can be more accurate than a wide range of previous template matching approaches.

This paper presents a new approach that goes further than the two above by learning a detector that directly takes advantage of the uniqueness of instances, and that does not need to be retrained or fine tuned in order to detect a new target object. This is done by learning an embedding that compares learned features of the target to learned features at each location in a scene image, and integrating this into a state-of-the-art detection framework.

The Target Driven Instance Detection problem is formulated as follows: given an input scene image \( S \) and a number, \( T \), of images of a target object, output a bounding box around the target object in \( S \), or no box if the object is not present. See Figure 1 for an example with \( T = 2 \) target images with the correct output shown.

The datasets [6], [7], [8] we use to evaluate our Target Driven Instance Detector (TDID) display a set of object instances in everyday home environments and exhibit real-world confounding factors such as scale variation, clutter and occlusion. An important aspect of the Active Vision Dataset [6] (used for test in many experiments) is that it was collected to sample views of household rooms from every position where a robot could navigate. As a result, many objects are quite small in some views, perhaps when seen a cross a room, and are partially occluded in many views.

The objects used in our experiments come from the BigBird and RGB-D Object datasets [9], [8], and we note that part of the methods success stems from seeing similar objects in training. This is the same in previous work to which we compare, and is reasonable to expect in real-world mobile manipulation applications, but it is important to make this clear.

We summarize our contributions as follows: (1) A novel

1University of North Carolina at Chapel Hill, Computer Science [ammirato, cyfu, mshevts, aberg]@cs.unc.edu
2 George Mason University, Computer Science kosecka@gmu.edu
Target Driven Instance Detector (TDID) model that easily transforms the current state-of-the-art general object detectors into instance detectors, as depicted in Figure 2. (2) Strong performance improvement on multiple challenging instance detection scenarios. (3) We compare TDID to previous work on one-shot training for instance classification, including more classic template matching, and show better accuracy. (4) The ability to generalize detection to unseen instances on challenging datasets without any additional training or fine-tuning.

II. RELATED WORK

Traditional methods for object detection in cluttered scenes follow the sliding window based pipelines where efficient methods for feature computation and classifier evaluation were developed such as DPM [10]. Examples of using these models in the table top setting include [11], [12]. Object detection and recognition systems that deal with textured household objects such as Collet et al. [13] and Tang et al. [14] take advantage of the discriminative nature of local descriptors. A disadvantage of these local descriptors is that they usually perform poorly in the presence of non-textured objects. Some of these issues were tackled by [15] which used template based methods to deal with such texture-less objects. Hand engineered features typically work well in table top settings that contain a relatively small number of objects at relatively large scale [16]. The authors in [17] introduce an effective approach to feature learning for simultaneous categorization and pose estimation for single objects on uniform backgrounds.

General Object Detector State-of-the-art object-category detectors have been improved significantly over the last few years in both accuracy and speed. These detectors rely on a backbone architecture, such as VGG [18] or ResNet [19], to extract features from the image, and then add a detection module on top of these features. Two-stage detectors, such as Faster R-CNN [1], and R-FCN [20], rely on an initial region proposal followed by a classification and location regression of the proposed regions. Recent single-stage detectors: YOLO [21], YOLOv2 [22], and SSD [2] skip the feature pooling stage and show that fast inference speed can be achieved. Recent work has added top-down connections [23], [24], [25], which can borrow rich semantic information from deeper layers and improve in accuracy for small objects, though usually at reduced speed.

Instance Recognition Compared to object-category recognition, the specific instance recognition setting has less intra-class variation and, is often allowed a limited number of training examples. Much work has been done using hand-crafted features and template matching to identify object instances even since somewhat recent seminal work [26], [27]. More recently hand-crafted features have been replaced with learned ones in many recognition tasks [28]. The instance recognition dataset BigBIRD [9], which provides dense, individual scans of over 100 object instances on a turntable has enabled more research on instance recognition. [5] shows that pre-training on BigBIRD improves robustness to pose and improves classification performance over hand-crafted and template matching methods, even if only one image per instance is provided for training.

Instance Detection The recent release of larger scale instance detection datasets like the Active Vision Dataset(AVD) [6] and GMU Kitchens [7], has enabled more work using deep learning for instance detection. The GMU Kitchen Dataset has 6,728 images across 9 scenes, and the initial release of AVD has 17,556 images across 9 scenes. Both datasets feature instances very similar to those in BigBIRD [9], with GMU featuring 11 such instances and AVD 30. [3], [4] attack the problem of limited training examples by synthesizing new examples with different background images. In both of these works general object category detectors such as SSD [2] or Faster R-CNN [1] are still used to solve the instance detection problem.

Navigation [29] address a related problem, exploring an environment to reach a target position. They also input both a target image (of the desired view) and an image from the current position, and learn an embedding to aid in navigation. It is not straightforward to adapt their method to the instance detection problem, however, as they aim to move so that the image at the current position matches the target exactly. The embedding is not designed to localize objects, which is necessary for detection. Furthermore, the network requires scene-specific layers, while most object detectors are expected to generalize to unseen environments.

Tracking Given an initial bounding box of an object, the tracking task is to localize the same object appearing in each subsequent video frame. Correlation is frequently used for estimating similarity of patches between frames [30] [31] [32]. Recent deep learning methods, such as [33], uses a Siamese network to measure the similarity
in tracking. [34] uses a correlation filter to transform the Siamese network to be fully convolutional. [35] combines the features of crops from previous and current frames to regress the location directly. [36] interprets the correlation filter learner as a differentiable layer and enables learning deep features that are tightly coupled to the correlation filter. Strong priors on the object and background exist in tracking, namely that neither changes much frame to frame. These priors include scale, location, illumination, orientation and viewpoint. Our instance detection setting requires robustness to larger changes between target and scene.

### III. Method

#### A. Problem Formulation

Instance detection requires a system to recognize and localize specific objects in novel images. Usually these scene images images contain many objects, some of which are instances to be recognized. Most object detectors work by training on a set of scene images and ground truth bounding boxes of objects, and test on novel scene images containing the same types of objects. General object detectors attempt to find all object instances in a scene image at once.

Our Target Driven Instance Detector (TDID), takes as input not only a scene image, but also one or more target images. These target images contain only the instance of interest, see Figure 1. TDID attempts to detect only this target instance in the scene image.

#### B. Network Architecture

TDID is similar to the Region Proposal Network (RPN), the first stage of Faster-RCNN, but adds a target/scene joint embedding. Figure 2 compares our architecture with that of the RPN. With this joint embedding we are able to outperform other detectors even without the second stage of the traditional Faster-RCNN pipeline. This results in an architecture that computes detection outputs in one shot, with speed close to other one-shot detectors, while achieving better accuracy on various instance detection tasks than both one and two stage detectors.

The high-level view of our architecture is as follows: Extract features from the target and scene images using some shared feature extraction network, such as VGG-16[18]. Next, pass both target and scene image feature maps through our joint embedding. Finally, a set of convolutions predict class scores and bounding box regression parameters for a set of default anchors boxes (see Figure 2) over the embedding feature map. In TDID there are only two classes: target object or background.

**Joint Embedding** We construct a joint embedding, see Figure 3 of all input images that can then be further processed for detection. The joint embedding combines feature correlation and differencing between the target image(s) and the scene image. The operations and features in the embedding are described below and Table I shows ablation results as different feature combinations are considered.

**Cross Correlation** is widely used in traditional methods with hand-crafted features for similarity matching. We started building the joint embedding by applying the cross correlation of target features with the scene features, generating a heatmap with only one channel dimension. This method generates a strong signal for predicting target presence/absence in each spatial location, but drops rich information from the feature channels. Depthwise-separable correlation applies correlation at each channel independently. This not only preserves more information for the subsequent instance localization but also yields high computational efficiency [37], [38]. We use depthwise-separable correlation in our joint embedding, represented as $CC$ in the ablation study, Table I and the green box in Figure 3.

**Feature differencing** is another way to measure similarity. Intuitively, a network attempting to learn a similarity between image features may do something like learn to subtract them. Instead of adding extra complexity to our framework by learning a similarity, we compute the difference directly and feed it as a signal to our joint embedding. We first apply global max pooling on the target features to bring them to $1 \times 1$ spatial resolution. Then we subtract this vector from each spatial location of the scene features. This feature is represented as $DIFF$ in the ablation study, Table I and the purple box in Figure 3.

**Scene Image Features** The features of the scene image from the feature extractor may also provide useful information for object detection. In the original RPN of Faster-RCNN, these are the features that are used to predict bounding boxes and potential objects. This feature is represented as $IMG$ in the ablation study, Table I and the white box in Figure 3.

**Ablation Study** We run an ablation study to show how using different combinations of features in our joint embedding affects detection performance. Results are reported for the instance detection task on split 2 of AVD. As expected, using just scene image features, $IMG$, fails as there is no information about the target instance. Surprisingly, using just $DIFF$ features provides a strong signal resulting in high detection performance. The addition of $CC$ features provides a small boost in performance here, and also proved to be useful in later experiments so it is included in our final model. $IMG$ features do not provided much new information from $DIFF$ and $CC$, while adding extra complexity and parameters to the network and so are omitted from our final model.

| Features Used | extra small | small | medium | large | All |
|---------------|-------------|-------|--------|-------|-----|
| $IMG$         | 1.9         | 7.7   | 5.3    | 5.3   | 2.2 |
| $CC$          | 23.8        | 58.5  | 44.0   | 50.7  | 27.7|
| $DIFF$        | 48.0        | 74.6  | 72.3   | 73.2  | 52.6|
| $IMG+CC$      | 28.0        | 54.5  | 51.4   | 54.8  | 51.9|
| $IMG+DIFF$    | 46.2        | 79.2  | 72.5   | 71.3  | 50.9|
| $CC+DIFF$     | 50.3        | 78.2  | 75.1   | 78.2  | 58.8|
| $IMG + CC + DIFF$ | 48.4     | 83.0  | 73.8   | 77.1  | 53.3|

**TABLE I**

Ablation study of features in TDID embedding on various object sizes in AVD split 2 [6]. $IMG==SCENE IMAGE FEATURES$, $CC==CROSS-CORRELATION$, $DIFF==DIFFERENCE$. MAP reported.
We first evaluate our system on a challenging object instance detection dataset, AVD [6]. We
use two target images (provided on the dataset website) for each instance, picking views to maximize how much of the object is seen. See Figure 1 for an example of two target images. We choose two target images because in general it may be impossible to recognize an instance from the back if only the front view is provided.

We report results for all three train/test splits reported in [6]. For training, we resize all images to 960x540 and use a learning rate of .001, momentum as .9 and weight decay 0.0005 and train for 40 epochs with . We then reduce the learning rate by a factor of 10, and continue training for another 15 epochs. Table IV shows that our method outperforms SSD[2], [6] on this task consistently, over 14 mAP on each split on all boxes (boxes > 50 x 30).

To produce a TDID system that runs at the same frame rate as SSD, we resize all images during testing to 720x405. We test the same model that was trained on the 960x540 images, and show results in the TDID(720x405) row in Table IV. We can see TDID still outperforms SSD by an average of over 5 mAP on all objects, and over 20 mAP on larger objects. We expect training a model at this resolution could result in even greater accuracy gains, while maintaining speed.

2) GMU Kitchens to AVD: We now compare on a different object instance detection task to Faster-RCNN [1]. [3] explore how to create synthetic training data for instance detection, and evaluate how their synthetic data can improve a detector’s performance when trained on one dataset, but tested on another. They train/test on the six instances present in both the GMU Kitchens dataset and AVD. In this task, the detector is trained on the GMU data, and tested on all images in the initial release of AVD (17,556 images).

First, we train only on the real images from GMU, and test on AVD. We use the same training hyper-parameters as in the previous instance detection task. On this challenging task TDID is able to outperform Faster-RCNN by over 8 mAP.

Next, we add synthetic images to training. [3] did not release their synthetic images, but did release code to generate them. We use their code and settings described in the paper to generate 5,160 synthetic images (31 report generating about 6000). Given extra training data, both Faster-RCNN and TDID improve. TDID retains its advantage over Faster-RCNN by 6 mAP, which may be further improved with better synthetic data.

B. One-Shot Instance Classification

We have shown our method outperforms state-of-the-art general object detectors on multiple instance detection tasks. We now show that we can also surpass other instance recognition and template matching work, as well as generalize to unseen target instances. [5] classify images of instances when given only a single image in training. They show a neural network, combined with some multi-view pre-training, can outperform previous non-deep-learning feature matching methods. [5] use a CaffeNet[48] classification network, pre-trained on ImageNet[49]. They then perform a multi-view pre-training step on BigBIRD, train on a single example of each instance in the RGB-D Scenes [8] dataset, and test classification accuracy on crops of instances in RGB-D Scenes.

We adapt our object detection framework to perform classification, and evaluate this modified network on the same one-shot instance classification task. In this setting, the definition of “target image” stays the same, but “scene image” is now a classification style image, i.e. a crop around one object. For TDID to generalize to unseen target instances it must be provided with a large variety of target instances during training. We construct a training set consisting of over 250 instances from the BigBIRD dataset and RGB-D Object Dataset, carefully excluding any instances that overlap with those in the test set, RGB-D Scenes.

For a fair comparison, we use AlexNet [28] (extremely similar to CaffeNet[48], same performance on ImageNet.

| Method | Box Size | Split 1 | Split 2 | Split 3 |
|--------|----------|---------|---------|---------|
| SSD[6] | 39       | 55      | 53      |         |
| TDID(720x405) | > 100 x 50 | 65.6    | 71.6    | 72.1    |
| TDID(960x540) | 70.3   | 75.4    | 72.7    |         |
| SSD[6] | 26       | 41      | 42      |         |
| TDID(720x405) | > 50 x 30 | 35.8    | 42.7    | 48.2    |
| TDID(960x540) | 48.9   | 55.8    | 56.5    |         |

TABLE IV

INSTANCE DETECTION RESULTS (mAP) ON THE AVD DATASET, WITH VGG16 BACKBONE.

| Train set | Method | Box Size | Real + | Synthetic |
|-----------|--------|----------|--------|-----------|
| Images    | RCNN   | 57.4     | 34.5   | 43.5      |
| Real      | TDID   | 75.8     | 73.8   | 43.3      |
| Real      | Faster | 69.9     | 44.2   | 51.0      |
| Synthetic | Faster | 57.7     | 42.1   | 50.9      |
| Synthetic | TDID   | 69.1     | 46.9   | 69.7      |

TABLE V

DETECTION PERFORMANCE (AVERAGE PRECISION) WHEN TRAINING ON GMU KITCHENS AND TESTING ON AVD. *SYNTHETIC IMAGES USED IN [3] AND OURS ARE SLIGHTLY DIFFERENT.

| Method | Accuracy |
|--------|----------|
| Random | 0.3      |
| BRISK  | 9.4      |
| ORB    | 6.6      |
| SURF   | 10.8     |
| BOLD   | 7.4      |
| SIFT   | 12.9     |
| Line-2D| 9.0      |
| Color Hist | 9.2 |
| HMP    | 25.4     |
| CaffeNet | 41.0    |
| CaffeNet+MV | 44.1 |
| TDIDours | 50.5    |

TABLE VI

ONE-SHOT INSTANCE CLASSIFICATION IN A SCENE.
Table VII

Few-shot detection average precision on GMU Kitchens.

| object                  | TDID | Faster-RCNN[3] |
|-------------------------|------|----------------|
| coca cola               | 30.8 | 88.5*          |
| coffee mate             | 73.8 | 95.5*          |
| honey bunches           | 52.0 | 94.1*          |
| hunt’s sauce            | 24.1 | 88.1*          |
| mahatma rice            | 26.7 | 90.3*          |
| nature v1               | 86.1 | 97.2*          |
| nature v2               | 82.2 | 91.8*          |
| palmolive orange        | 25.3 | 80.1*          |
| pop secret              | 62.2 | 94.0*          |
| pringles bbq            | 26.0 | 92.2*          |
| red bull                | 37.9 | 65.4*          |
| mAP                     | 48.2 | 88.8*          |

classification) pre-trained on ImageNET as our backbone network. To test how well our system can generalize to unseen target instances, we do not train on the single example of each test instance as [5] does. Instead we use the provided example as the target image at test time, never re-training our network or updating the weights to recognize these new objects. Even without any fine-tuning on the test objects our method achieves 50.5% classification accuracy, outperforming the previous deep-learning approach that does train on the test objects, as well as several feature and template matching methods. See Table VI.

C. Few Shot Instance Detection

We next explore few-shot instance detection with two examples of each instance available, one front view and one back view. In contrast with usual few-shot tasks, we do not train on examples of test objects. We use the examples as target images at test time, requiring our detector to generalize to unseen objects without any on-line training or fine-tuning. High performance on this task could be useful for many applications where the system is given just a few examples of a target object but does not have time to re-train. We test on the instances in the GMU Kitchens dataset.

As in the previous experiment, to enable TDID to generalize we constructed a training set with many different target instances, from AVD, BigBIRD, RGB-D Objects, RGB-D Scenes, and ImageNET VID. VID consists of snippets of video with one or more objects labeled with a bounding box. While training TDID, we first choose a video at random, then choose an object as the target, and crop two random frames of the video to get target images. Another frame from the video is chosen as the scene image 50% of the time, while a frame from a different video is chosen the other 50%. This means the target object is visible in the scene image in half of the examples. We use the same instances from BigBIRD and RGB-D Objects as in Section IV-B, but instead of cropped classification images we use code from [3] to synthetically place the objects in 1449 images from NYUD2[50].

In addition to the released bounding boxes in AVD, we take advantage of the dataset’s structure to add more target instances automatically. Starting with an image, I, in a scene, S, we use selective search to get the bounding box of some object or region, O, in I. Using the camera locations and depth images provided by AVD, we can project O to world coordinates and then project back into every other image in S to get the bounding box of O in every image. This gives us more target instances almost for free. Unfortunately this setup is still experimental, and is not always robust to occlusion and other factors. Therefore we only generate samples from two scenes from AVD, adding 5000 target/scene image pairs. Future work includes making this process more robust to hopefully greatly increase TDID’s generalizability.

We use the same training hyper-parameters as in the detection experiment on AVD in Section IV-B, except we cut the learning rate in half to .0005 and train for 150,000 iterations.

As shown in Table VII, TDID is able to generalize well to these instances, achieving 48.2 mAP. We also provide the Faster-RCNN results [3] from training/testing on split one of the GMU data as a sort of upper bound reference, and to show the difficulty of the GMU data relative to other tasks. This result is particularly exciting as TDID is able to give reasonable performance on a task general category detectors cannot perform. The ability to detect novel objects quickly, without any new training, could be very valuable for robots in many applications.

V. Analysis, Limitations, and Future Work

TDID is able to outperform previous object recognition systems at a variety of challenging tasks. Given the comparison of a target to a potential detection, the remaining classification problem is binary, matching the target or not. Category detection methods need to discriminate between many different classes, both requiring more training data per class, and often a second stage network (e.g. in the Faster RCNN, consisting of RPN followed by classification and final bounding box regression). Our method is based on the idea that it is sometimes easier to learn to compare two things, than to learn about every object. This may be especially relevant when a small number or only one training example is used as shown in the results in Table 6 where TDID outperforms a wide range of methods on single shot classification.

Our method still has the same difficulty of detecting small objects as other detectors, as can be seen in the drop in performance as box size changes in Table IV. In addition, while TDID’s ability to detect objects it has never seen during training (Table VII) is exciting, the objects it can generalize to are still limited. All of the train/test objects came from a similar household/grocery store domain.

Future work includes improving the generalizability and detection performance of TDID to objects across many different domains. We hope in the future an object detection system will be able to work off the shelf, detecting any objects a robot or other system may be interested in without the need for training.
