Robust Object Estimation using Generative-Discriminative Inference for Secure Robotics Applications

(Invited Paper)

Yanqi Liu, Alessandro Costantini, R. Iris Bahar
Brown University
School of Engineering
Providence, RI

Zhiqiang Sui, Zhefan Ye, Shiyang Lu, Odest Chadwicke Jenkins
University of Michigan
Dept. of Computer Science and Engineering
Ann Arbor, MI

ABSTRACT

Convolutional neural networks (CNNs) are of increasing widespread use in robotics, especially for object recognition. However, such CNNs still lack several critical properties necessary for robots to properly perceive and function autonomously in uncertain, and potentially adversarial, environments. In this paper, we investigate factors for accurate, reliable, and resource-efficient object and pose recognition suitable for robotic manipulation in adversarial clutter. Our exploration is in the context of a three-stage pipeline of discriminative CNN-based recognition, generative probabilistic estimation, and robot manipulation. This pipeline proposes using a SAmpling Network Density filter, or *SAND filter*, to recover from potentially erroneous decisions produced by a CNN through generative probabilistic inference. We present experimental results from SAND filter perception for robotic manipulation in tabletop scenes with both benign and adversarial clutter. These experiments vary CNN model complexity for object recognition and evaluate levels of inaccuracy that can be recovered by generative pose inference. This scenario is extended to consider adversarial environmental modifications with varied lighting, occlusions, and surface modifications.

KEYWORDS

Robust machine learning, Robot perception, DNN adversarial attack

ACM Reference Format:
Yanqi Liu, Alessandro Costantini, R. Iris Bahar and Zhiqiang Sui, Zhefan Ye, Shiyang Lu, Odest Chadwicke Jenkins. 2018. Robust Object Estimation using Generative-Discriminative Inference for Secure Robotics Applications: *(Invited Paper)*. In *IEEE/ACM INTERNATIONAL CONFERENCE ON COMPUTER-AIDED DESIGN (ICCAD '18), November 5–8, 2018, San Diego, CA, USA*. ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3240765.3243493

ICCAD '18, November 5–8, 2018, San Diego, CA, USA

ACM ISBN 978-1-4503-5950-4/18/11…$15.00
https://doi.org/10.1145/3240765.3243493

1 INTRODUCTION

Even for common human environments, we are still far from accurate, reliable, and resource-efficient object and pose recognition suitable for dexterous robotic manipulation. Convolutional neural networks (CNNs) for object recognition have recently gained widespread use in robotics for scene understanding [42], dexterous object manipulation [13], autonomous driving [1], and a plethora of compelling applications. While demonstrating high accuracy, such CNNs incur the cost of vastly complex parametric models with high energy consumption profiles. Highly parallel and GPU computing methods have been proposed to meet the needs of CNNs computation given the relatively unbounded resources of desktop workstations and cloud computing.

Unfortunately, for efficiency, CNN computation has proven less amenable to the resource budgets available to embedded platforms and onboard processing for autonomous robots. The high computational and energy cost of CNNs is assumed to be due to complexity of their network architectures. In particular, this complexity grows with the number of layers and weight parameters in the network, as well as computational operations used in training and inference.

The reliability problems robots face for object and pose recognition become that much more challenging when an adversary can modify the environment to exploit the vulnerabilities of a CNN. A possible malicious attack has the potential to drastically alter (and perhaps manipulate) the final behavior of a robotic system. In computer vision, Szegedy *et al*. [37] have shown that there are many malicious techniques that create adversarial examples. Slight modifications of an original image, often not perceptible to the human eye, can be detrimental to neural network performance. Continued work in computer vision [2, 3, 17, 19, 28] has further shown such malicious modifications to be relatively easy to realize with the capacity to drastically change the recognition result.

In the context of robot manipulation, an adversary may not be able to directly alter images observed by a robot, but can alter the environment from which the robot is capturing its image observations. The clutter that naturally occurs in common human environments can be enough to defeat the recognition abilities of a CNN. Similar to adversarial image manipulation, natural clutter could also be slightly and maliciously altered to deceive a CNN used for robot perception. Examples of such alterations include moving an object to create an occlusion, modifying the appearance of an objects surface, or dimming the room lights.

In this paper, we investigate factors for accurate, reliable, and resource-efficient object and pose recognition suitable for robotic

manipulation in adversarial clutter. Our exploration is in the context of a three-stage pipeline of discriminative CNN-based recognition, generative probabilistic estimation, and robot manipulation.

Within this three-stage pipeline, we posit that accurate, reliable, and resource-efficient robot perception could be met through generative-discriminative approach to inference. This idea more specifically performs generative probabilistic inference for second-stage estimation using CNN results from first-stage estimation. The central principle of this pipeline is to filter decisions produced by the CNN through probabilistic inference before making hard, and potentially erroneous, decisions. Embodied by our previously proposed SAmpling Network Density filter, or *SAND filter* [36], this form of inference should provide robustness to CNN errors and adversarial alterations, explainability over probable interpretations of a scene, and greater ability to tune for computational efficiency. Second-stage generative inference in the SAND filter essentially serves as a recovery mechanism from first-stage CNN errors that could occur, both as false positives and false negatives. For efficiency, this ability to recover offers the prospect of using smaller and more lightweight CNN architectures that trade-off accuracy, for a significant reduction in computation and energy consumption. For malicious alterations, the ability to recover could also improve recognition robustness to CNN errors due to deception.

Towards this end, we present experimental results from SAND filter perception for robotic manipulation in tabletop scenes with both benign and adversarial clutter. These experiments vary CNN model complexity for first-stage object recognition and evaluate levels of inaccuracy that can be recovered by second-stage pose inference. These results compare the SAND filter approach with a baseline using Faster-RCNN object detection followed by Iterative Closest Point (ICP) pose estimation, as a simple pose estimation algorithm. This scenario is extended to consider environmental modifications by an adversary. These results provide initial insights into robot perception and manipulation for malicious scenes with varied light conditions, increased the level of occlusion, and alteration of the face of an object.

2 RELATED WORK

2.1 Object detection

Recent studies have shown that convolutional neural networks are powerful tools for vision based object detection. The classical object detection framework, Regional Convolutional Neural Network (RCNN) [10], is a two stage model, combining convolutional layers to extract the features of proposed bounding boxes and a linear classifier to classify the object. Fast RCNN [9] improves the runtime of the object detector by creating a single-stage end-to-end training model. Faster RCNN [30] improves the performance even further by adding two extra convolutional layers as a Region Proposal Network (RPN) that shares convolutional layers with the object detection network and reduces the cost of computing regional proposals. The YOLO approach [29] gets rid of the regional proposal pipeline in all RCNN works and presents a single pass network that generates bounding boxes and object prediction for each feature. However, it fails to achieve comparable accuracy as faster RCNN. The most recent detection network, Mask-RCNN [43], is an extension of Faster RCNN that adds pixel-to-pixel alignment and a binary mask to represent object spacial layout, which greatly improves the detection accuracy. However, Mask-RCNN requires that the training images be segmented labels. There has been other research concerning the datasets used to train the CNN for object detection. The works of Lai *et al*. [20, 21] and Silberman *et al*. [25] use RGBD datasets that contain fairly small amounts of training data comparing to ImageNet. These representative datasets resemble of the scale of training assumed in this paper. These works show that we do not necessarily need a large dataset to achieve high accuracy. Given the small size of our current training dataset and the relatively high accuracy of its object detector, we choose to use faster-RCNN in our baseline approach.

Long *et al*. [22] propose fully convolutional networks (FCN) for semantic segmentation by replacing fully connected layers in traditional CNN with 1×1 convolutional layers. FCNs take images of arbitrary size and provide per-pixel classification labels. However, FCNs are not able to separate neighboring objects within the same category to obtain instance-level labels; hence we cannot directly re-task FCN for object detection purposes. Nonetheless, most unified approaches are based on FCN to localize and classify objects using the same networks. Recently, there has been a trend to utilize FCN to perform both object localization and classification [31], [30] [29], [16]. Sermanet *et al*. propose an integrated CNN framework for classification, localization and detection in a multiscale and sliding window fashion [31]. Morris *et al*. [23] propose a fully-convolutional Pyramid Network in operate at successive resolutions as information flows up the pyramid to the lowest resolution. In our work, the input to our SAND filter's CNN stage is a pyramid of images with different scales in order to generate a heatmap for the second stage. To perform object detection we replace fully connected layers with convolutional layers.

2.2 Robot object manipulation

Reliable operation of autonomous mobile manipulators remains an open challenge for robotics, where perception remains a critical bottleneck. Within the well-known sense-plan-act paradigm, truly autonomous robot manipulators need the ability to perceive the world, reason over manipulation actions afforded by objects towards a given goal, and carry out these actions in terms of physical motion. However, performing manipulation in unstructured and cluttered environments is particularly challenging due to many factors. Particularly, to execute a task with specific grasp points demands first recognizing object and estimating its precise pose.

For object and pose estimation, PR2 interactive manipulation [4] segments non-touching objects from a flat surface by clustering of surface normals. This work uses RGBD data from cameras that provide both color and depth values at every pixel. Similarly, Collet *et al*.presented a discriminative approach, MOPED, to detect object and estimate object pose using iterative clustering-estimation (ICE) using multiple color cameras [5]. Narayanan *et al*. [24] integrate A* global search with the heuristics neural networks to perform scene estimation from RGBD, assuming known identification of objects. Papazov *et al*. [27] used a bottom-up approach of matching the 3D object geometries using RANSAC and retrieval by hashing methods.

Deep learning on RGBD has also been applied to robotic grasp detection through deep reinforcement learning, such as work by

Gualtier *et al*. [12], and supervised shape completion, such as by Varley *et al*. [41]. For manipulation in cluttered environments, Ten Pas *et al*. have shown success detecting viable grasp poses in RGBD point clouds using geometric inference [39] and estimation by deep neural networks [38]. Sui *et al*. [35] built on these methods to recognize objects as well as graspable poses, in order to perform purposeful goal-directed manipulation. This work combines the output of discriminative inference methods, such as CNNs, with probabilistic generative inference to improve robustness. While demonstrating effectiveness, the methods above use neural network architectures, such as VGG16, that is expensive in both computation and energy. These models also assume improved recognition accuracy directly implies improvement in robot manipulation, which needs greater validation experimentally.

2.3 Adversarial attacks

Szegedy *et al*.[37] demonstrated that adversarial examples are misclassified by different classifiers both in the case of different architectures or different subsets of the training data[17] [2][40]. These results are confirmed also in cases where the differences between these examples were indistinguishable to the human eye. Kurakin *et al*. [19] confirmed the results in a simple physical scenario. Papernot *et al*. [28] showed a case of a black-box attack against a neural network, where adversaries have no knowledge about the model. Others works proposed a possible solution during the training phase: Panda *et al*. [26] proposed to inject random noise on the training data and Zheng *et al*. [44] presented a stability training method to avoid mis-prediction due small input distortion.

Defending against adversarial examples is hard because it requires neural network models to produce good outputs for every possible input; however, in a physical scenario neural networks must work well only on a very small number of all the many possible inputs. In this work, in order to deal with adversarial scenarios, we chose to not modify the initial training set, thereby avoiding an excessive human effort during the data collection phase.

3 PROBLEM DESCRIPTION

Our casting of the robotic manipulation problem takes as input RGBD observations from the robot's camera and outputs a motion trajectory for robot execution of a manipulation action. The resulting motion trajectory is assumed to be executed by a low-level motion controller. This execution is considered successful if the object acted upon ends up in a desired position and orientation, within some given tolerance. The robot manipulation problem has three stages: detection of relevant objects, pose estimation for these objects, and generation of the manipulation motion trajectory. For object detection, inputs are given as an RGB observation z that views a scene of k^* relevant objects. The first stage of the process will estimate this scene as a collection of k objects. Each object i will have an estimated object label o_i and 2D image-space bounding box b_i. Object labels are strings containing a semantic identifier of the object's class, assumed to be a human-intuitive reference. In the second stage, a six degree-of-freedom object pose q_i will be estimated for each object in the frame relative to the robot's camera. We assume that every object is independent of all other objects. Thus, the state of an individual object i in the scene is represented as $x_i = \{q_i, b_i, o_i\}$.

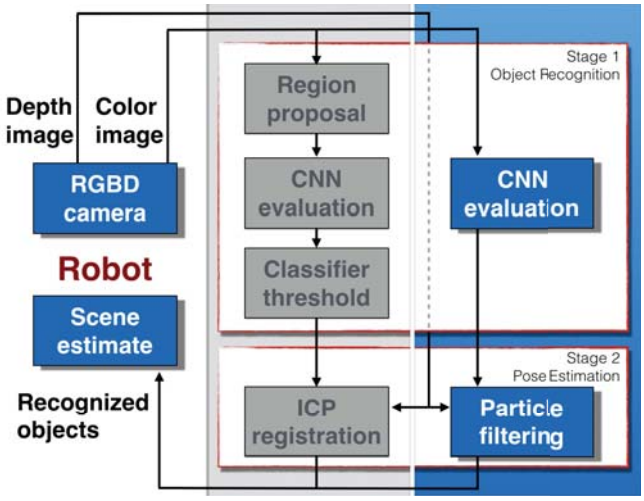


Figure 1: Framework overview of *baseline* approach and *SAND filter* approach. Both approaches aim to recognize objects and their poses from a robot's color and depth sensing. The flow in the grey background is the *baseline* approach which performs the hard thresholding in the first stage and uses the ICP algorithm for the second stage. In contrast, the *SAND filter* approach avoids making a hard threshold in the object detection stage so that the generative sampling method can explore over a larger state space.

In the third stage, manipulation actions u are computed to move a particular object j from its estimated pose q_j to a desired goal pose q_j^G. This third stage forms as a sequence of robot configurations u that will be used to control the robot's actuators, assuming low-level motor control. The formation of these trajectories for u are mostly considered through the invocation of one of many possible motion planning algorithms [34]. The perception problem we focus on this this paper only considers the first two of these three stages.

4 METHODS

In this section, we first discuss the baseline approach composed of Faster-RCNN plus a simple pose estimation method. Next, we describe the *SAND filter* approach composed by a pyramid-CNN followed by a sampling-based local search method. Both approaches are two-stage methods where the first stage performs object detection and the second stage performs pose estimation. However, instead of making hard decisions in both stages as is done in the baseline approach, the SAND filter approach avoids the need for hard thresholding in the first stage, which allows for a more robust estimation process in the second stage. Figure 1 describes the flow of the two approaches.

4.1 Faster-RCNN and Simple Pose Estimation

We implemented Faster-RCNN detection across a set of CNN architectures of varying complexity. The purpose of Faster-RCNN is to return the evaluation of bounding box regions for all objects in

a color image from a robot view. Each evaluation returns the confidence the object observed in the bounding box that belongs to a particular object class.

Faster-RCNN uses the Regional Proposal Network (RPN), a fully convolutional network that can predict object locations, represented as bounding boxes. RPN is followed by a classification network that shares extracted features from the convolutional layers. The classification network predicts object classes using the softmax function given the bounding boxes. Between RPN and a classification network, there is a Region of Interest (ROI) pooling operation for feature extraction from RPN. We adopt the pooling operation (ROIAlign) from [43] since it preserves exact spatial location.

In our work, we implement Faster-RCNN with a number of variations across the VGG[33], ResNet[14], and AlexNet[18] network architectures. The network configuration is shown in Table 1. For the VGG network, we use all 5 levels of the convolutional layers to extract image features for RPN and the rest of the fully connected layers as the classification network to generate object classes from the proposed region. For ResNet, we use the first 3 residual block layers that share convolution features with RPN and the last residual layer as the classification network. For AlexNet, we use all 5 convolutional layers for RPN and all 3 fully connected layers as the classification layer.

Table 1: Faster RCNN network configuration

Network	RPN	Classification Network
AlexNet	5 conv layers	3 FC layers
VGG	all conv layers	3 FC layers
ResNet	first 3 residual layers	last residual layers

The purpose of the second stage is to estimate the object pose, q_i, given a bounding box, b_i, and an object label, o_i, from the first stage. By extracting the corresponding point cloud of b_i and the object geometry of o_i, Iterative Closest Point (ICP) is then used to estimate the pose, q_i.

ICP is a discriminative method that minimize the energy between two sets of points sets iteratively. The energy term is defined as the sum of Euclidean squared error between two point clouds. During each iteration, ICP will select two subsets of points using RANSAC [8] from their corresponding point clouds to calculate the energy. In this work, we adopt the off-the-shelf ICP implementation from Point Cloud Library (PCL)[1]. Due the nature of our scene, we initialize the object pose prior to ICP by computing the centroid position of the cropped point cloud. Once the energy between two point sets is lower than the convergence criterion, we obtain a rotation, R, and translation, t, that represents the transformation from the initial pose to the final object pose, q_i.

4.2 SAmpling Network Density (SAND) Filter

Sui et al. [36] propose the two-stage SAND filter approach for robust object detection along with accurate 6DoF pose estimation. The SAND filter approach builds upon both CNN and generative sampling-based search methods for sampling the network density. Unlike the Faster-RCNN used in the baseline approach for the first stage, the pyramid-CNN is proposed in the SAND filter approach to avoid hard thresholding in the first stage. In addition, the generative sampling method for pose estimation in the second stage can take full advantage of the probability density provided by pyramid-CNN.

The output of the pyramid-CNN is a pyramid of heatmaps which serves as proposals to the second stage. Each pixel in the heatmap represents a bounding box with a categorical distribution of all object classes and each heatmap corresponds with a fixed shape of bounding box. As there are no hard thresholdings in the pyramid-CNN, the number of proposals for the second stage is much more than the thresholded detection results from the Faster-RCNN in the baseline approach. Although more proposals lead to more false positives, the second stage can be better informed to find objects which are originally false positives in the baseline approach. In the work by Sui et al. [36], the fully connected layers in VGG-16 are replaced with 1×1 convolutional layers to perform object detection in each window location. In this paper, we try the same set of network architectures in pyramid-CNN and replace the fully connected layers with convolutional layers.

The generative sampling-based search in the second stage for the pose estimation of the SAND filter is inspired by sampling methods, such as the boostrap filter [11]. This method takes a full probability density from the pyramid-CNN and performs a search over the pose state space weighted by the density prior. Given an object class, a collection of samples used to represent pose state hypotheses are first initialized by importance sampling over the pyramid heatmaps. The weight of each sample is a linear combination of the object classification confidence from the first stage and the geometric confidence. The geometric confidence is evaluated by comparing the cropped observation point cloud with the rendered point cloud given the pose hypothesis in the sample. The comparing function counts the number of points in these two point clouds match each other. After the weights evaluation, a resampling and diffusion process is performed over the collection of samples. The search process of evaluation-resampling-diffusion repeats until convergence of the object pose.

5 ANALYSIS

In this section, we first describe the performance metrics for first and second stage. We then compare the performance between the *baseline* approach and the *SAND filter* approach. We choose to analyze AlexNet as well as various versions of VGG, and ResNet for our network benchmarks. All the networks are pre-trained on the ImageNet dataset [6] and then fine-tuned on our dataset, which contains 15 household objects. The object detector is implemented in PyTorch and trained on an Nvidia Titan Xp.[2]

5.1 Performance Metrics

To compare the detection results, we adopt the evaluation framework from the Pascal VOC Challenge [7]. Each detection output, which consists of 2D coordinates with a confidence score, is assigned to a ground truth bounding box. If the overlapping area, or intersection over union (IoU), between the detection and ground truth is over 0.5, we consider the detection as a match, or true positive (TP), otherwise,

[1] http://pointclouds.org/

[2] http://pytorch.org/

it is a false positive (FP). If no detections satisfy the overlapping criterion with the ground truth, it becomes a false negative (FN). We further calculate the average precision for each object class in our dataset. Average Precision (AP) is the mean precision given different recall thresholds, which describes the shape of the PR curve. Precision is defined as $\frac{TP}{TP+FP}$, and recall is defined as $\frac{TP}{TP+FN}$. We can then obtain one AP for each object class, and the mean average precision (mAP) of all object classes.

For evaluating pose accuracy, we adopt the metric from [15] to measure the mean of the pairwise point distance from two sets of point clouds. This metric can also account for symmetric objects, which is useful in comparing daily household objects.

5.2 Object Detection (First Stage) Accuracy

Table 2: Object detection (first stage) accuracy among different networks with progress dataset

Network	Faster-RCNN(mAP) (baseline)	Pyramid-CNN(mAP) (SAND filter)
AlexNet	0.860	0.327
VGG11	0.885	0.246
VGG13	0.915	0.279
VGG16	0.926	0.296
VGG19	0.896	0.247
ResNet18	0.903	0.225
ResNet34	0.914	0.188
ResNet50	0.919	0.242
ResNet101	0.897	0.236
ResNet152	0.924	0.217

Table 2 reports the accuracy achieved by the object detector implemented with various CNN models from the first stage of Faster-RCNN and Pyramid-CNN. We can see that for the object detection stage of Faster-RCNN, more complex networks do not necessarily generate better results than smaller networks, even though more complex networks can produce less prediction error on ImageNet. For example, VGG16 has higher mAP accuracy than VGG19 and similarly ResNet50 has higher mAP than ResNet101. This result is due in part to the limited size of our training dataset. The dataset we used to finetune the network is small compared to ImageNet and also category specific (i.e., 15 household objects). This means a lot of detectors in more complex models are not learning meaningful features of the objects and thus not contributing to the overall detection accuracy. As mentioned in Section 2.1, we do not necessarily need a large dataset to acheive high accuracy. However, a small dataset does not necessarily indicate we should always choose a small network.

For the Pyramid-CNN, we can see that the performance on object detection is much lower than the Faster-RCNN since Pyramid-CNN does not perform hard thresholding for the object detection stage. That is, it generates many more detection outputs that lead to more false positives. We used offline hard example mining [32] to reduce some of these false positives. Hard example mining effectively takes the falsely detected images during the validation phase, explicitly creates negative examples out of these images, and adds the negative examples to the training set.[3] However, while we still end up with more false positives (even with hard example mining), by not thresholding, our approach makes the generative sampling in the second stage of the *SAND filter* explore and search more thoroughly in order to correct false detections. As we will demonstrate in Section 6, this also makes our approach less susceptible to mistakes from conditions that mimic adversarial attacks.

5.3 Pose Estimation (Second Stage) Accuracy

Table 3: Pose estimation (second stage) accuracy among different networks using a distance threshold of 0.1m.

Network	ICP Acc(%) (baseline)	Particle Filtering Acc(%) (SAND filter)
AlexNet	0.662	0.788
VGG11	0.707	0.742
VGG13	0.687	0.753
VGG16	0.702	0.758
VGG19	0.707	0.783
ResNet18	0.697	0.727
ResNet34	0.697	0.646
ResNet50	0.692	0.727
ResNet101	0.697	0.712
ResNet152	0.697	0.682

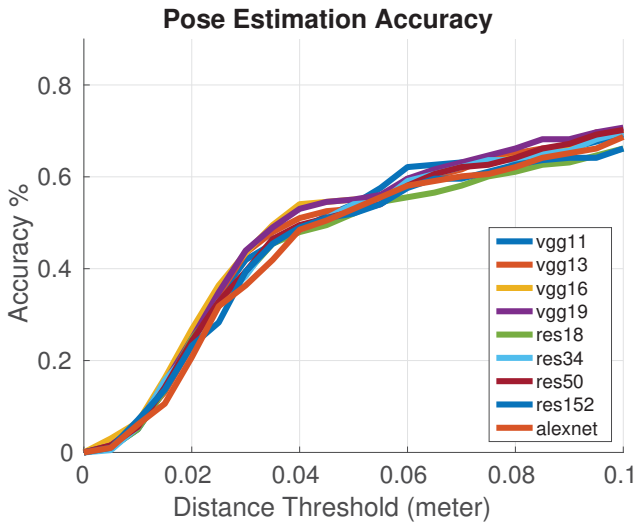


Figure 2: The final pose estimation accuracy of the *baseline* approach composed of Faster-RCNN and ICP. The y-axis is the accuracy percentage based on different distance threshold on x-axis.

[3] Note that this extra hard example mining step does not have any effect on the accuracy of Faster-R-CNN since it performs thresholding. It is therefore not used for the baseline experiments.

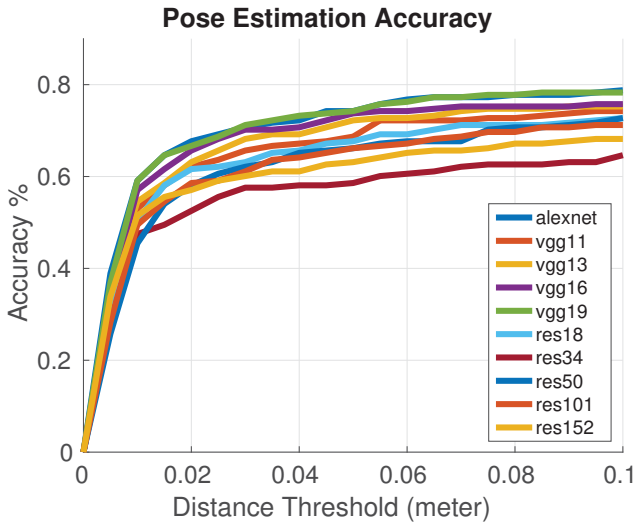


Figure 3: The final pose estimation accuracy of the *SAND filter* approach composed of Pyramid-CNN and particle filtering. The y-axis is the accuracy percentage based on different distance threshold on x-axis.

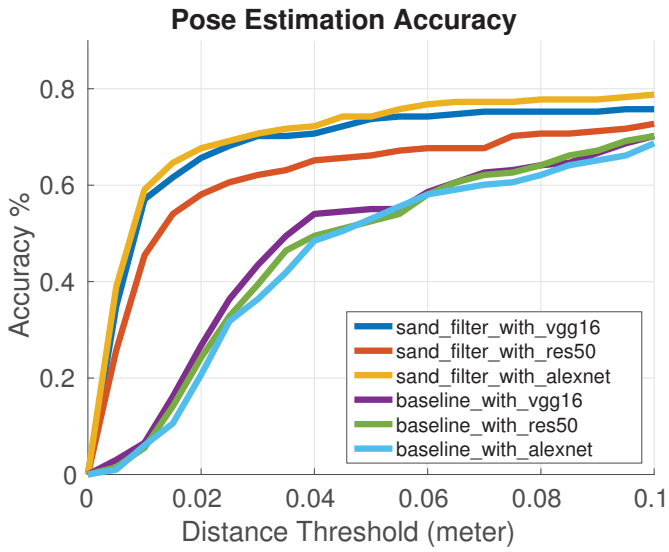


Figure 4: Pose accuracy of the *SAND filter* approach comparing with *baseline* approach.

Figures 2 and 3 show the accuracy vs. distance threshold results for the pose estimation stage given the first stage results generated from the various CNN models. Table 3 shows the numerical accuracy of the second stage using a distance threshold of $0.1m$. Note that direct comparisons between pose estimation accuracy values and object detection accuracy values are not meaningful since these values are derived in different ways.

However, for the *baseline* approach, we can see that lower first stage accuracy does not necessarily mean that this will lead to lower second stage accuracy. For example, VGG11 and VGG16 have significant accuracy differences in the first stage, but achieve the same pose estimation accuracy in the second stage. These results demonstrate that the object detection accuracy should not be the sole predictor of accuracy in the pose estimation stage. Using smaller CNNs for detection can generate comparable result as using bigger networks.

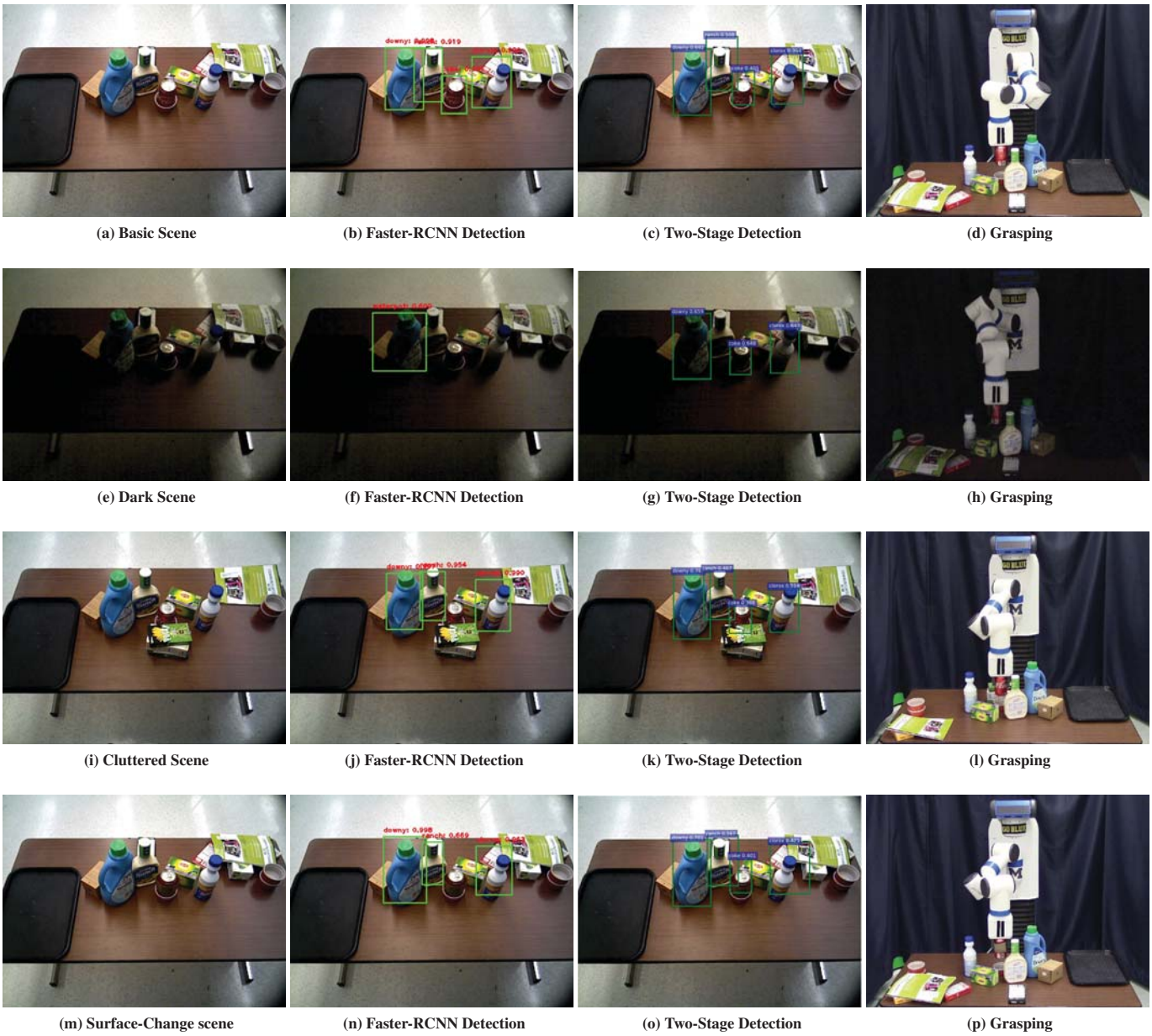
The same conclusion can be applied to the *pyramid-CNN* used in the SAND filter approach. AlexNet has a simpler network than VGG; however, its pose estimation performance is better. In general, we observe that for *pyramid-CNN* a higher first stage accuracy can help to reach a higher second stage accuracy, but it is not always necessary. For example, VGG19 and VGG11 have lower first stage accuracy than VGG16, but they have same or better second stage accuracy.

Shown in Figure 4, we compare the pose estimation accuracy of the baseline and SAND filter approaches using AlexNet, VGG16, and ResNet50 for the object detection stage. We see that the *SAND filter* approach is consistently and significantly better than the *baseline* approach within a tighter distance threshold, e.g. 0.02 meter. For the 0.1 meter threshold, *SAND filter* approach is still better than the *baseline* approach.

6 ROBOTIC EXPERIMENT

In this experiment, we compare the robot grasping results using Faster-RCNN and our SAND filter method for the object detection and pose estimation stages to get an initial understanding of how these two approaches might perform under adversarial attack. The task for the robot is to recognize the Coke can and then pick and place it on the black tray. Due to limited time and space, it was possible to perform the robot experiment with only one architecture, for either Faster-RCNN or Pyramid-CNN. Hence, our choice was to use VGG16 because it is the CNN architecture with the highest object detection accuracy and nearly highest pose estimation accuracy for Faster-RCNN, giving the baseline the best chance to succeed with the adversarial example.

Figure 5 shows the results from these two methods. We first make a basic scene with normal light condition and minimum occlusion of objects in our dataset ("Coke", "clorox", "downy" and "ranch"). We then mimic three adversarial scenes by changing the light conditions, making the Coke can partially occluded and altering the surface of the Coke. In the basic scene, both Faster-RCNN and our two-stage method correctly recognize all the objects and the robot successfully picks and places the Coke on the tray. However, in the adversarial scenes, the performance of Faster-RCNN is not satisfactory while the two-stage method remains reliable. In the dark scene, Faster-RCNN only detects one object, but it is a false positive, while our SAND filter method finds three objects correctly though missing the "ranch". When the Coke is partially occluded or wrapped with a cover, Faster-RCNN fails to detect it, while the SAND filter approach not only gives the correct label and bounding box but also a good pose estimation, so that the robot succeeds in picking and placing.



(a) Basic Scene (b) Faster-RCNN Detection (c) Two-Stage Detection (d) Grasping

(e) Dark Scene (f) Faster-RCNN Detection (g) Two-Stage Detection (h) Grasping

(i) Cluttered Scene (j) Faster-RCNN Detection (k) Two-Stage Detection (l) Grasping

(m) Surface-Change scene (n) Faster-RCNN Detection (o) Two-Stage Detection (p) Grasping

Figure 5: The robot task is to pick the can of Coke and place it on the black tray under various adversarial scenes. One basic scene (a) and three adversarial scenes (b)(c)(d) are shown above. (b) is a dark scene, (c) is a cluttered scene in which the Coke is partially occluded, and (d) is a scene in which the Coke is wrapped with a brown cover. Figures (b)(f)(j)(n) show the detection results from the Faster-RCNN detector with a threshold of 0.5. The Faster-RCNN detector was not able to detect the Coke in those adversarial scenes. Figures (c)(g)(k)(o) show the detection results from the two-stage SAND filter method. The detector missed the bottle of ranch in the dark scene, but successfully detected all the Coke cans and other objects in our dataset. Figure (d)(h)(i)(p) show the moment when the robot successfully picked up the Coke can, given the detection from the SAND filter method.

7 CONCLUSIONS

In this paper, we explore the role of convolutional neural networks for robust robot manipulation, where the CNN is used for object detection in the first stage of a three stage process. Furthermore, we explore how a generative sample approach for pose estimation in the second stage can improve performance of the robot. In particular, we show that the accuracy gains of increasing network complexity of the CNN used in the first stage may not be necessary to obtain

high accuracy. Furthermore, we show that relying solely on the CNN to make hard decisions for scene perception tasks leaves open several challenges and vulnerabilities, especially when dealing with complex scenes (e.g., with non-ideal lighting or clutter). By using a two-stage SAND filter process that avoids hard thresholding of the CNN output and instead performs generative sampling on a heat map of object proposals, we demonstrate its promise in providing robust robot manipulation for complex scenes. We posit that these complex scenes may also have some similarities to scenarios used for malicious attacks. Future work will include a more thorough investigation of adversarial attack strategies, how they may alter scenes, and how generative-descriminative approaches may be used to more robustly handle these attacks.

REFERENCES

[1] Mariusz Bojarski, Davide Del Testa, Daniel Dworakowski, Bernhard Firner, Beat Flepp, Prasoon Goyal, Lawrence D. Jackel, Mathew Monfort, Urs Muller, Jiakai Zhang, Xin Zhang, Jake Zhao, and Karol Zieba. 2016. End to End Learning for Self-Driving Cars. *CoRR* abs/1604.07316 (2016). arXiv:1604.07316 http://arxiv.org/abs/1604.07316

[2] Nicholas Carlini and David Wagner. 2016. Towards evaluating the robustness of neural networks. *arXiv preprint arXiv:1608.04644* (2016).

[3] Yu-Hsiu Chen, Ting-Hsuan Chao, Sheng-Yi Bai, Yen-Liang Lin, Wen-Chin Chen, and Winston H Hsu. 2015. Filter-invariant image classification on social media photos. In *Proceedings of the 23rd ACM international conference on Multimedia*. ACM, 855–858.

[4] Matei Ciocarlie, Kaijen Hsiao, Edward Gil Jones, Sachin Chitta, Radu Bogdan Rusu, and Ioan A Şucan. 2014. Towards reliable grasping and manipulation in household environments. In *Experimental Robotics*. Springer Berlin Heidelberg, 241–252.

[5] Alvaro Collet, Manuel Martinez, and Siddhartha S Srinivasa. 2011. The MOPED framework: Object recognition and pose estimation for manipulation. *The International Journal of Robotics Research* (2011), 0278364911401765.

[6] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*. IEEE, 248–255.

[7] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. 2010. The pascal visual object classes (voc) challenge. *International journal of computer vision* 88, 2 (2010), 303–338.

[8] Martin A Fischler and Robert C Bolles. 1987. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. In *Readings in computer vision*. Elsevier, 726–740.

[9] Ross Girshick. 2015. Fast R-CNN. *arXiv preprint arXiv:1504.08083* (2015).

[10] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jagannath Malik. 2014. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*. IEEE, 580–587.

[11] Neil J Gordon, David J Salmond, and Adrian FM Smith. 1993. Novel approach to nonlinear/non-Gaussian Bayesian state estimation. In *IEE Proceedings F (Radar and Signal Processing)*, Vol. 140. IET, 107–113.

[12] Marcus Gualtieri, Andreas ten Pas, and Robert Platt Jr. 2017. Category Level Pick and Place Using Deep Reinforcement Learning. *CoRR* abs/1707.05615 (2017). arXiv:1707.05615 http://arxiv.org/abs/1707.05615

[13] M. Gualtieri, A. ten Pas, K. Saenko, and R. Platt. 2016. High precision grasp pose detection in dense clutter. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 598–605. https://doi.org/10.1109/IROS.2016.7759114

[14] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 770–778.

[15] S. Hinterstoisser, V. Lepetit, S. Ilic, S. Holzer, G. Bradski, K. Konolige, , and N. Navab. 2012. Model Based Training, Detection and Pose Estimation of Texture-Less 3D Objects in Heavily Cluttered Scenes. (2012).

[16] Lichao Huang, Yi Yang, Yafeng Deng, and Yinan Yu. 2015. Densebox: Unifying landmark localization with end to end object detection. *arXiv preprint arXiv:1509.04874* (2015).

[17] Sandy Huang, Nicolas Papernot, Ian Goodfellow, Yan Duan, and Pieter Abbeel. 2017. Adversarial attacks on neural network policies. *arXiv preprint arXiv:1702.02284* (2017).

[18] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*. 1097–1105.

[19] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. 2016. Adversarial examples in the physical world. *arXiv preprint arXiv:1607.02533* (2016).

[20] Kevin Lai, Liefeng Bo, and Dieter Fox. 2014. Unsupervised feature learning for 3d scene labeling. In *Robotics and Automation (ICRA), 2014 IEEE International Conference on*. IEEE, 3050–3057.

[21] Kevin Lai, Liefeng Bo, Xiaofeng Ren, and Dieter Fox. 2011. A large-scale hierarchical multi-view rgb-d object dataset. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*. IEEE, 1817–1824.

[22] Jonathan Long, Evan Shelhamer, and Trevor Darrell. 2015. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 3431–3440.

[23] Daniel D Morris. 2018. A Pyramid CNN for Dense-Leaves Segmentation. *arXiv preprint arXiv:1804.01646* (2018).

[24] Venkatraman Narayanan and Maxim Likhachev. 2016. Discriminatively-guided Deliberative Perception for Pose Estimation of Multiple 3D Object Instances. In *Proceedings of Robotics: Science and Systems*. AnnArbor, Michigan. https://doi.org/10.15607/RSS.2016.XII.023

[25] Pushmeet Kohli Nathan Silberman, Derek Hoiem and Rob Fergus. 2012. Indoor Segmentation and Support Inference from RGBD Images. In *ECCV*.

[26] Priyadarshini Panda and Kaushik Roy. 2018. Explainable Learning: Implicit Generative Modelling during Training for Adversarial Robustness. *arXiv preprint arXiv:1807.02188* (2018).

[27] Chavdar Papazov, Sami Haddadin, Sven Parusel, Kai Krieger, and Darius Burschka. 2012. Rigid 3D geometry matching for grasping of known objects in cluttered scenes. *The International Journal of Robotics Research* (2012), 0278364911436019.

[28] Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z Berkay Celik, and Ananthram Swami. 2017. Practical black-box attacks against machine learning. In *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security*. ACM, 506–519.

[29] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. 2015. You only look once: Unified, real-time object detection. *arXiv preprint arXiv:1506.02640* (2015).

[30] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. 2015. Faster R-CNN: Towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems*. 91–99.

[31] Pierre Sermanet, David Eigen, Xiang Zhang, Michaël Mathieu, Rob Fergus, and Yann LeCun. 2013. Overfeat: Integrated recognition, localization and detection using convolutional networks. *arXiv preprint arXiv:1312.6229* (2013).

[32] Abhinav Shrivastava, Abhinav Gupta, and Ross B. Girshick. 2016. Training Region-based Object Detectors with Online Hard Example Mining. *CoRR* abs/1604.03540 (2016). arXiv:1604.03540 http://arxiv.org/abs/1604.03540

[33] Karen Simonyan and Andrew Zisserman. 2014. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556* (2014).

[34] Ioan A. Şucan, Mark Moll, and Lydia E. Kavraki. 2012. The Open Motion Planning Library. *IEEE Robotics & Automation Magazine* 19, 4 (December 2012), 72–82. https://doi.org/10.1109/MRA.2012.2205651 http://ompl.kavrakilab.org.

[35] Zhiqiang Sui, Zheming Zhou, Zhen Zeng, and Odest Chadwicke Jenkins. 2017. SUM: Sequential scene understanding and manipulation. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 3281–3288. https://doi.org/10.1109/IROS.2017.8206164

[36] Zhiqiang Sui, Zhefan Ye Zhou, and Odest Chadwicke Jenkins. 2018. Never Mind the Bounding Boxes, Here's the SAND Filters. *arXiv preprint arXiv:1808.04969* (2018).

[37] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. 2013. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199* (2013).

[38] Andreas ten Pas, Marcus Gualtieri, Kate Saenko, and Robert Platt. [n. d.]. Grasp pose detection in point clouds. *The International Journal of Robotics Research* ([n. d.]), 0278364917735594.

[39] Andreas Ten Pas and Robert Platt. 2016. Localizing handle-like grasp affordances in 3d point clouds. In *Experimental Robotics*. Springer, 623–638.

[40] Matej Uličnỳ, Jens Lundström, and Stefan Byttner. 2016. Robustness of deep convolutional neural networks for image recognition. In *International Symposium on Intelligent Computing Systems*. Springer, 16–30.

[41] J. Varley, J. Weisz, J. Weiss, and P. Allen. 2015. Generating multi-fingered robotic grasps via deep learning. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 4415–4420. https://doi.org/10.1109/IROS.2015.7354004

[42] Yu Xiang, Tanner Schmidt, Venkatraman Narayanan, and Dieter Fox. 2018. PoseCNN: A Convolutional Neural Network for 6D Object Pose Estimation in Cluttered Scenes. *Robotics: Science and Systems (RSS)* (2018).

[43] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. 2017. Aggregated residual transformations for deep neural networks. In *Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on*. IEEE, 5987–5995.

[44] Stephan Zheng, Yang Song, Thomas Leung, and Ian Goodfellow. 2016. Improving the robustness of deep neural networks via stability training. In *Proceedings of the ieee conference on computer vision and pattern recognition*. 4480–4488.