CURRICULUM VITAE

Name:	Justin D. Rokisky
Degree and date to be conferred:	Master of Science, 2020
Collegiate institutions attended:	University of Maryland, Baltimore County, MS in Computer Science, 2020 Rutgers University, New Brunswick, BA in Computer Science, 2012
Field of study:	Computer Science
Professional positions held:	Deep Learning Research Intern, The Johns Hopkins Applied Physics Lab, Laurel, MD Summer 2020 Software Engineer, Mindgrub Technologies, Baltimore, MD, August 2016 - May 2020 Middle School Math Teacher, Baltimore City Public Schools, Baltimore, MD, Aug 2014 - August 2016

ABSTRACT

Title of Thesis: Using Web Images & Natural Language for Object Localization in a Robotics Environment

Justin D. Rokisky, Master of Science, 2020

Thesis directed by: Dr. Cynthia Matuszek
Department of Computer Science

The ability for humans to interact with robots via language would allow for more natural interactions between robots and humans. To this end, in this work I introduce a novel approach that allows robots to localize objects from an unbounded set of classes given only a description of a target object. The first part of this work is a performance analysis of current state of the art object detectors and a region proposal approach [1] on the Autonomous Robot Indoor Dataset [2]. The second part of this work introduces a three stage natural language guided webly object localization approach and associated experiments to evaluate its performance. The first stage of the approach generates a webly dataset without any manual curation from a human description of the target object. The second stage of the approach uses the webly dataset to train a binary classifier for the target object. Finally, region proposals from selective search [1] are input to the webly supervised binary classifier and the region proposal with the highest confidence score is returned as the prediction.

Using Web Images & Natural Language for Object Localization in a Robotics Environment

by

Justin D. Rokisky

Thesis submitted to the Faculty of the Graduate School of the
University of Maryland, Baltimore County, in partial fulfillment
of the requirements for the degree of
Master of Science
2020

Advisory Committee:
Dr. Cynthia Matuszek, Chair/Advisor
Dr. Frank Ferraro
Dr. Tim Finin

Acknowledgments

I would like to thank Dr. Cynthia Matuszek for advising me through the production of this thesis and during my time at UMBC. I would also like to thank Dr. Frank Ferraro and Dr. Tim Finin for taking the time to participate as members of my thesis defense committee. I would like to thank Pat Jenkins for his general guidance while working on this thesis and Erebus Oh for the work she did on finding and providing code for commercial computer vision services. Finally I would like to thank my wife Samantha for her continuous support throughout my time pursuing a graduate degree and throughout every other aspect of my life.

Table of Contents

List of Tables

List of Figures

List of Abbreviations

ARID	Autonomous Robot Indoor Dataset
ARID WOD	Autonomous Robot Indoor Dataset Web Object Dataset
BOVW	Bag of Visual Words
CNN	Convolutional Neural Network
HRI	Human Robot Interaction
ILSVRC	ImageNet Large Scale Visual Recognition Challenge
IOU	Intersection Over Union
PHOG	Pyramid of Histogram of Oriented Gradient
PHOW	Pyramid of Histogram of Visual Words
RGB-D	Red Green Blue Depth
ROD	RGB-D Object Dataset
RPN	Region Proposal Network
SVM	Support Vector Machine

Chapter 1: Introduction

The promise of robots operating alongside humans continues to move closer towards reality. One promising area of human robot interaction (HRI) is the development of systems that allow humans to communicate with robots via natural language as this will allow for easier and more natural interactions [3]. In this work I introduce a novel approach that allows robots to localize objects from an unbounded set of classes given only a description of a target object. The approach requires no ground truth data or knowledge of target classes as it uses an oracle, in this case internet search engines, to ground a given object description to example images of the target object which are then used to train a deep-learning based classifier.

One challenge associated with robotic vision object detection and localization is the necessity of being able to handle the dynamic nature of human environments. As many state of the art deep learning object detection and localization approaches are trained on datasets with a fixed number of classes [4–6] they are unable to identify objects outside of these classes. It is possible to increase the number of available classes by manually annotating data for the new classes and retraining classifiers, but this is an intensive task. To remedy this, work has been done [2, 7–11] towards utilizing the web as a source of weakly supervised training data. Approaches that

use web data in such a way are generally referred to as *webly* supervised approaches [7,12,13]. There are difficulties associated with using web data for training classifiers such as handling noise in web images [14–16] and dealing with differences between the web domain and target domain [10, 13]. The approach in this paper tries to overcome these difficulties and the additional difficulties associated with not having a bounded and known set of object classes by utilizing a region proposal ranking system. Since the target object is guaranteed to be present in the scene, even if the learned webly representation is weak due to noise or domain differences, the region containing the target object should be ranked the highest due to it being the most similar to the representation learned from the web data.

The main contribution of this thesis is the introduction of a three stage natural language guided webly supervised object localization approach for use in a robotic environment with novel objects. A description of each stage of the approach follows. The first stage of the approach involves collecting a short natural description of an object in the robot's environment. This short description is then used to query image search engines to generate a webly dataset of images that match the given description. The second stage of the approach uses the gathered web data and MS COCO [6] crops to train a binary classifier for the target object. Finally, selective search [1] is used to generate region proposals which are input to the binary classifier and then ranked according to their confidence scores. The highest ranking region proposal is returned as the location of the target object.

Chapter 2: Related Work and Background

2.1 Related Work

In this section I describe work that relates to this thesis. I specifically focus on robotic vision object detection, supervised object detection, webly supervised vision, and grounded language.

2.1.1 Robotic Vision Object Detection

In this section I provide an overview of some robotic vision RGB-D datasets that can be used for testing robotic vision object detection and localization approaches as well as some approaches that have been performed on these datasets. Some of the unique challenges faced in robotic vision that necessitate the construction of specific datasets are the occlusion of objects, varied lighting conditions, the presence of novel objects, and the relative placement of the sensor(s) being used for data collection.

Due to the varied tasks of robotic platforms (e.g. navigating spaces, grasping objects), the presence of sensors from multiple modalities can be a big benefit. To this end, the availability and effectiveness of commercial RGB-D cameras (e.g.

Xbox Kinect, Intel Realsense) have led to them being a mainstay in robotics vision research [2, 17–20]. The RGB-D Object Dataset (ROD) [17] was one of the earlier robotic vision RGB-D datasets released. It contains both multi-view RGB-D images of 300 houshold objects from 51 categories as well as RGB-D video sequences taken in human environments (e.g. a kitchen) of some of those object instances. The RGB-D video sequences were later extended in [18]. Other RGB-D robotic vision datasets have focused on objects in more natural human environments under varied operating conditions [2, 19, 20].

Multiple object detection and object classification approaches have been attempted on the previously mentioned datasets [17]. In older works, classical computer vision approaches such as using sliding windows coupled with handcrafted features and SVMs [17] have been used for object detection. More modern approaches that do not use handcrafted features have also been introduced for robotic vision [18]. There has also been work done on the other parts of the object detection pipeline, such as a 3D multi-view region proposal method [19] and enhancing robotic vision datasets with synthetic data [21] and web images [2].

This work builds on previous work in robotic vision object detection in two ways: I show the performance of state of the art and commercial object detectors on a robotic vision dataset and following in the footsteps of Georgakis et al. [19], show the performance of selective search on a different robotic vision dataset.

2.1.2 Supervised Object Detection

As many of the current state of art object detection approaches are supervised learning based, they require large datasets consisting of cleaned and labeled data to perform well. These datasets are important as they are used as benchmarks to compare the performance of different approaches. Some of the key object detection benchmark datasets are ILSVRC [5], MS COCO [6] and Pascal VOC [4]. Generally object detection approaches can be split into two different categories [22]: single stage approaches [23–29] and two stage approaches [30–34]. With the introduction of DETR [35], there is now a third category: transformer based approaches.

Single stage approaches generate object predictions with no intermediate step. Compared to two stage approaches, they tend to have faster inference times but worse accuracy [36]. The YOLO family [25–27] are popular examples of single stage approaches. Yolo [25] introduced an extremely fast CNN based object detector that works by splitting the input image into grids and having each grid predict the conditional class probability of an object and bounding boxes with associated confidence scores. One of the limitations of YOLO is that it struggles with predicting multiple objects in close relation to one another due to the limited number of bounding boxes predicted per grid cell. Building on [25], Redmon et al. introduced YOLOv2 [26]. They noted that two areas YOLO struggled with were lower recall than two stage approaches and a greater number of localization errors and in response make multiple changes such as: introducing batch normalization, multi-scale training, a pass-through feature layer, etc. In [27], Redmon and Farhadi introduced

YOLOv3 which improved upon YOLOv2 [26]. As noted in the title of the paper, "YOLOv3: an Incremental Improvement", incremental changes such as: increasing the depth of the feature extraction network, adding bounding box predictions across 3 scales, and using multilabel classification for the grid cell bounding box prediction are made to increase performance while maintaining fast inference speed.

Another single stage approach, SSD, was introduced by Liu et al. in [23]. The SSD network utilizes the early layers of standard image classification networks (e.g. [37]), chops off the classification layers, and adds additional convolutional layers to allow for multi-scale feature maps for object detection. They report inference times faster than YOLO while also being more accurate than Faster R-CNN. Noticing that the class imbalance between foreground and background examples during training was one of the key reasons why single stage approaches struggled to compete with two stage approaches, in [24] Lin et al. develop a *focal loss* for down-weighting the loss assigned to well-classified examples (e.g. easy background cases). This focal loss is used in RetinaNet, a single stage object detector that uses a feature pyramid network [38] on top of a standard convolutional network as its backbone and then has two subnetworks per feature pyramid layer: one for classification and one for bounding box regression.

Two stage approaches split the object detection process into two steps: generating region proposals and evaluating those proposals. One of the first two stage object detectors to use deep learning based approaches was R-CNN [30] by Girschick et al. The R-CNN architecture consists of three modules: a region proposal network , a CNN for feature extraction (they use AlexNet [39]), and linear SVMs (one per

class) to classify the extracted features. As each module in R-CNN needs to be trained and run separately this has a large negative impact on performance and in [32] Girschick introduces Fast R-CNN, which unifies the latter two portions of R-CNN to improve performance. First the per-proposal feature extraction is replaced by a process that extracts features from the entire image once and then region of interest pooling is used to generate feature maps per region proposal. Second, each region's feature map is input into a subnetwork that predicts both the softmax class probabilities and class specific bounding box regression. This network is trained using a multi-task loss. Girschick reports up to an 18x training speedup and up to 213x test speedup over R-CNN. After Fast R-CNN replaced the latter two modules of R-CNN, in [33] Ren et al. introduced Faster R-CNN which utilized a deep learning based region proposal network (RPN) to replace the region proposal module. The RPN uses the same whole image feature extraction as the rest of Fast R-CNN so additional computation is limited. The RPN is trained with the Fast R-CNN module in an approximate joint fashion ensuring that neither diverges from the other. The authors report much faster inference time than Fast R-CNN with higher accuracy.

The transformer based approach introduced by Carion et al. in [35] is the first transformer based object detection approach. The architecture of DETR consists of a CNN backbone for feature extraction, a transformer encoder that processes the extracted feature along with a positional encoding, a transformer decoder that processes the encoder's results along with object queries, and finally a feed forward network that uses the output of the decoder to predict a bounding box or no object.

The authors report performance similar to an optimized version of Faster R-CNN.

The approach introduced in this thesis is inspired by the two stage approaches. Like R-CNN and Fast R-CNN, it uses selective search [1] as the region proposal method but any region proposal method could be used. The two key differences between the approach in this paper and other object detection approaches are that it is not trained on a finite set of object classes, as it uses human descriptions coupled with the web to generate training data, and that it leverages the knowledge that the described object exists in the given image. This means that the approach only needs to find the regions in the image that are the closest to the learned representation of the target object and not some absolute representation of the object.

2.1.3 Webly Supervised Vision

There are a very large amount of images coupled with descriptive information on the internet that have been indexed by search engines (e.g. Google, Bing) and made available via commercial sites (e.g. Flickr). As many of these search engines and sites are searchable, it has become relatively easy to gather many different images of object instances. That being said, there are associated difficulties in using web images such as noise and differences between the web domain and target domain. Some datasets [4, 5] use images from web sources, but the image selection process is heavily curated. The WebVision dataset [40] was introduced as a dataset for comparing the performance of webly approaches.

One of the biggest difficulties in using webly datasets for training learning

approaches is the amount of noise in web images [14–16]. One specific type of noise is the presence of outlier images. In [41] a one-class SVM is used to remove such outlier images from webly datasets. Critical of using outlier detection approaches for removing noise in webly datasets, Smyth et al. [42] introduce an approach that focuses on removing elements that will have negative training value from the webly dataset. Leveraging the presence of image metadata, Yang et al. [43] analyze image metadata to remove noisy outliers. Instead of removing noisy webly dataset elements, curriculum based approaches [7, 44, 45] rank elements by the amount of noise/complexity they contain and then train on easier elements first followed by the more difficult elements.

Another difficulty associated with using webly datasets for training learning approaches is bridging the gap between the web domain and the target domain. In [13], Tao et al. introduce a webly supervised object detector the uses a domain adversarial loss to adapt the web domain to the target domain. In [10], Bergamo et al. introduce a domain adaptation approach that when given some number of high quality supervised data examples can use web examples to train an image classifier.

Beyond the straightforward approach of using multiple search engines [11] and querying for the target object name, other complex strategies have been introduced for generating webly datasets. One approach is to use query expansion [11, 46, 47] to increase the quality of images that are returned from search engines. Another type of approach involves using the reverse image lookup feature of search engines [12, 41, 48] to find either descriptive information (e.g. object name) about the target image or related images. Related images are directly used as the webly dataset and in the

case of descriptive information, it is processed and used as the input to generate a webly dataset.

Due to the dynamic nature of robotic environments, effective webly supervised vision approaches would be a huge benefit to the robotics community. There have been multiple attempts to use web image data for image classification in a robotics context [2,11,12,41,49] but generally classifiers trained on webly data do not perform as well as those trained on non-webly data. The approach introduced in this work differentiates itself from other webly supervised robotic vision approaches utilizing human descriptions of objects to generate a webly dataset. The human description allows for a more descriptive query string to be used in the process of searching for web images.

2.1.4 Grounded Language

Grounding language is the process of mapping natural language to representations [3,50,51] of objects [52–55], concepts [56–59], or actions [60–65], in the physical world. This is an important field for human robot interaction (HRI) as successful grounded language acquisition approaches can enable more natural interaction between humans and robots.

Work has been done to use language for robotic object detection [53–55]. Nguyen et al. introduce an approach for object retrieval based on a natural language request for an object that can be used to perform certain actions (e.g. to cut something) [54]. Krishnamurthy et al. introduce an approach for mapping nat-

ural language to target objects in a physical environment by learning how to map language to physical categories and relational phrases [55]. Hill et al. use episodic memory coupled with one-shot learning to allow an agent to learn to identify an object with only a single visual perception and description of the object [53].

The approach introduced in this work differentiates itself from other grounded language approaches by using search engines coupled with deep learning based approach to map a natural language description of an object to the object in the physical world.

2.2 ARID: Autonomous Robot Indoor Dataset

In this section I provide an in depth overview of ARID, the dataset that the introduced approach is evaluated on. In Loghami, Caputo, and Vincze's work Recognizing Objects In-the-wild: Where Do We Stand? [2], they noted the lack of a dataset for benchmarking robotic vision based object detection approaches in real-life environments. In response to this they introduce a new dataset for this purpose: the Autonoumous Robot Indoor Dataset. This dataset differs from most other robotic vision datasets in that it contains scenes with varied lighting, occluded objects, and from the perspective of a mobile robot in a natural human environment. The dataset consists of rgb images synced with their depth image counterparts and the coordinates of bounding boxes for the included object categories. The authors also introduced an auxiliary dataset called the ARID Web Object Dataset that consists of filtered web images for the object categories present in the main dataset.

2.2.1 Dataset Statistics

Although the paper mentions that the complete dataset contains 6,000+ images with 120,000+ bounding boxes, the publicly released dataset consists of 3,251 annotated images with 40,262 bounding boxes. The images are frames from 197 different video segments roughly equally split between each point in the environment. This restricted dataset was created by the authors to perform an image classification experiment comparing the performance of 5 different CNN architectures trained on the ARID object crops, the University of Washington's RGB-D Object Dataset [17], and the ARID Web Object Dataset. To help work with this dataset programatically, a python package has been developed and is available at: https://github.com/Jrokisky/Arid-Dataset-Helper.

2.2.2 Collection System and Process

The dataset was collected in what appears to be a university lab environment under various lighting conditions from the perspective of a mobile robot with a commercial RGB-D camera, the Asus Xtion Pro. During each session, the robot would loop through four separate waypoints of which instances can be seen in 2.1.

Upon reaching each waypoint, the robot would horizontally pan its RGB-D camera to record the environment with a resolution of 640x480 pixels. After the robot performed two loops through the waypoints, the object instances were randomly moved. After the data collection process was completed, every fifth image was chosen to be part of the dataset and the RGB images were synced with their

Figure 2.1: Instances of RGB images from ARID [2] each of the four waypoints. The top left, top right and bottom left images are of the same table with different object instances and placements from different points of view. The location of the bottom right image can be seen in the background of the top right image by using the microwave as a reference. These images were used with the explicit permission of their creator.

Figure 2.2: 20 randomly selected object instances from the 51 available object categories in ARID [2]. These images were used with the explicit permission of their creator.

compliment depth images.

2.2.3 Object Categories

The dataset contains 153 object instances that are grouped into 51 different categories. These object categories are the same object categories in the University of Washington's RGB-D Object Dataset [17]. Bounding boxes are given for each object instance present in each image. Examples of some object instance crops can be seen in 2.2.

Chapter 3: A Three Stage Natural Language Guided Webly Supervised Object Localization Approach

The approach introduced in this thesis is a 3 stage natural language guided webly supervised object localization approach 3.1. The first stage uses a natural language description of a target object input into multiple search engines to generate a webly dataset of images of the target object. The second stage learns a representation from the webly dataset via a binary classifier. Although there may be noise in the webly dataset and domain differences between the web domain and target domain, salient and identifiable features should be shared between the majority of the images in the webly dataset and target object. The goal of the binary classifier is to learn to recognize these highly salient features. The third stage uses the learned binary classifier to rank region proposals generated by selective search [1] by their confidence scores. The highest scoring region will contain the strongest representation of the learned salient features which will hopefully map to the location of the target object. Therefore the highest scoring region is returned as the approach's prediction of the target objects location in the scene.

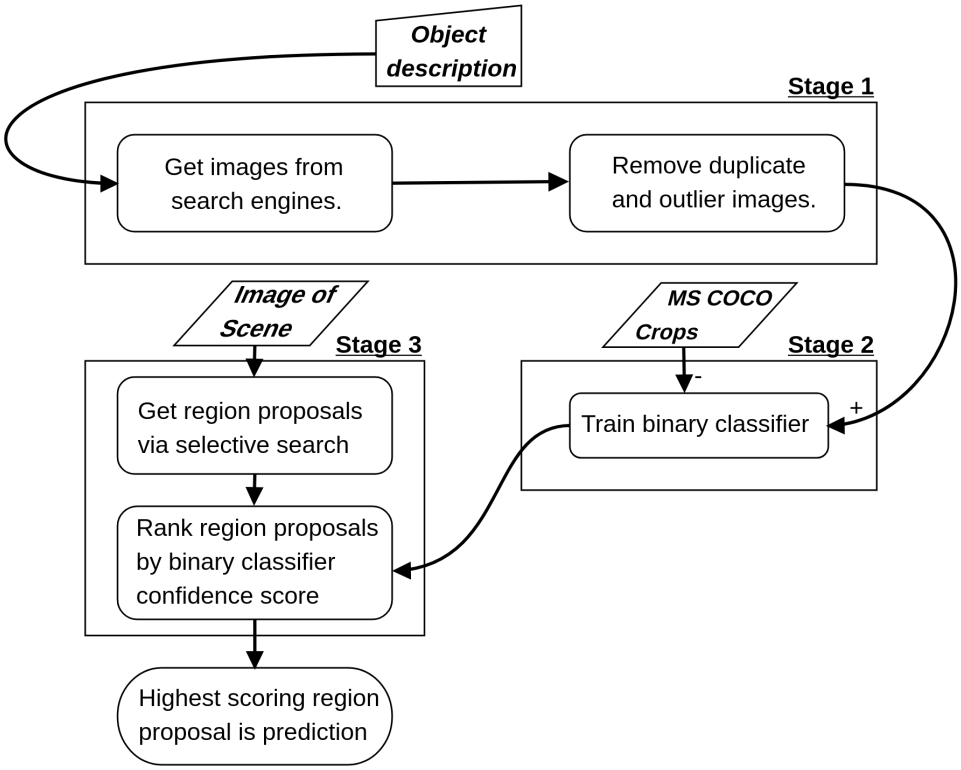


Figure 3.1: High level overview of the introduced approach.

3.1 First Stage: Automatically Generating a Webly Dataset

The first step in the introduced approach is to acquire a description of some object in the given scene. This description should ideally be short, informative, and contain the name of the object (e.g. yellow lemon fruit). This description is then input to multiple search engines (https://yandex.com/images/, flickr.com, and www.bing.com/images), as in [11] and [7], and up to 1000 image results are saved from each search engine. Next the web images are cleaned and prepared to be used in training. First, duplicate and outlier images are removed [11]. Images are considered outliers if their feature embedding (as computed by ResNet-152 [66] pretrained on ImageNet [5]) has an l2 distance from the mean that is in the top 30%. Each of these steps is done automatically with no manual curation or oversight.

3.2 Second Stage: Learning a Representation

After the webly dataset has been created it is combined with crops from the MS COCO [6] validation set to form a dataset for the training of a binary classifier. The images in the webly dataset are used as the positive class. The top 25 largest crops from each MS COCO [6] class are used as the negative class. tdAny collection of images of objects could be used as the negative class here. Ideally the negative class contains a diverse set of objects to ensure that the negative class is robust. The mean and standard deviation of the dataset are computed and used to normalize the dataset. Three other transforms are applied to the training dataset:

images are resized to 128 x 128, a random horizontal flip is applied [67], and random erasing [67, 68] is applied. The Resnet-152 [66] architecture, pretrained on ImageNet [5] and with the classification layer replaced with a single element linear layer, is finetuned for 10 epochs on the dataset. The dataset transforms and deep learning model were chosen after experimenting with different transforms and models on the validation dataset (the *lemon_0* object instance dataset). Precision-recall curves for each learned binary classifier applied to the ARID object instance crops can be seen in 4.3.

3.3 Third Stage: Generating and Processing Region proposals

Finally, the *fast* variant of selective search [1] is used to generate region proposals. The *fast* variant of selective search was chosen after experimentation 4.2 as it offered a reasonable trade-off between the number of proposed regions that would need to be processed and its ability to recall ARID object bounding boxes at a reasonable IOU threshold. The proposed regions are input into the learned binary classifier and ranked according to their confidence score. The highest scoring region is returned as the approach's predictiion for the location of the target object in the scene. Non-maximum suppression is applied when more than one proposed region is desired to ensure that different proposals encompassing the same object are not returned.

Chapter 4: Experiments and Analysis

In this chapter I present the experiments that were conducted to assess the viability of the approach introduced in this paper. I first analyze the performance of current state of the art and commercial service object detectors on a subset of ARID to understand how well they perform on a robotics dataset and to establish which model to use as a baseline to compare the introduced approach to. Then I analyze the performance of different variants of selective search [1], the region proposal method used in the introduced approach, on ARID to determine which variant to use. Next, I analyze performance of the learned binary classifiers used in the second stage of the introduced approach on ARID object crops. Then I analyze the performance of the entire introduced approach over multiple object instances in ARID. Finally, I compare the performance of the introduced approach to one of the best performing models, Faster R-CNN [33], on the subset of object instances that are present in both ARID and MS COCO. Aggregate results can be seen in 4.1.

4.1 Object Detector Performance

In this section, I analyze the performance of two different groups of object detection approaches on ARID: current state of the art object detectors and com-

mercially available proprietary services. The goal of this experiment is to find the model(s) that performs best, without finetuning/training, so they can be used as a baseline for evaluating the performance of the introduced approach. Since the current state of the art object detectors and potentially the commercially available services are deep learning based, one difficulty faced when comparing object detector performance is that the object categories they were trained to detect might differ from those present in ARID as well as from each other. To remedy this, I randomly choose 60 video segments (15 taken from each of the available waypoints) and randomly select one frame from each segment. After frames have been selected, I input these frames into each object detection approach and record the predictions with a confidence score of over 70%. I then manually inspect these predicted bounding boxes. For each predicted bounding box, I consider 4 possible bounding box prediction types: the bounding box does not include any visible object, the bounding box includes an entire object with an inaccurate label, the bounding box includes part of an object with an accurate label, and the bounding box includes an entire object with an accurate label. I also note when an object detection model does not correctly produce a bounding box over an object instance from a known object category. Object categories are considered known if we have access to the object categories that the approach was trained on or inferred from other seen predictions. Overlapping bounding boxes were removed.

4.1.1 State of the Art Object Detectors

In this section, I analyze the ability of three state of the art object detectors trained on MS COCO: Faster R-CNN [33], YoloV3 [27], and DETR [35] to detect MS COCO object categories in the selected ARID images. Object detection performance was manually evaluated as there were not object instances present from a majority of MS COCO object categories. These three models were chosen as they are representative of three different types of object detection approaches: a two stage network (Faster R-CNN), a single stage network (YoloV3), and a transformer based network (DETR). They all also have publicly available Pytorch [69] implementations, perform well on the MS COCO benchmark [22], and have publicly available MS COCO pretrained weights.

4.1.1.1 Shared ARID and MS COCO Object Categories

The MS COCO 2017 dataset consists of 80 object categories, 8 of which are explicitly shared with ARID. The shared object categories are: *apple*, *scissors*, *keyboard*, *orange*, *bowl*, *toothbrush*, *banana*, and *cell phone*. There are two MS COCO object categories, *cup* and *bottle*, that are not direct matches to ARID object categories, *coffee_mug* and *water_bottle*, but encompass those object categories so they were considered to be shared as well. As ARID was collected in a natural human environment, there are also several other MS COCO object categories present such as person, toothbrush, tv/monitor, microwave, and laptop.

4.1.1.2 Faster R-CNN

Faster R-CNN [33] is an object detection network that builds upon Fast R-CNN [32] by introducing a deep learning based region proposal network (RPN). The RPN generates regions that are then input to a Fast R-CNN network. The RPN shares convolution features with the Fast R-CNN network allowing region proposals to be generated at minimal cost which yields significantly faster inference performance compared Fast R-CNN. The RPN and Fast R-CNN networks are trained simultaneously.

The Faster R-CNN [33] implementation offered by the Pytorch Torchvision package available at https://pytorch.org/docs/stable/torchvision/models.html#faster-r-cnn was used to produce bounding box predictions. This implementation has a ResNet-50-FPN backbone and has been pretrained on the MS COCO 2017 training dataset. The publicly available MS COCO pretrained weights can be found at https://download.pytorch.org/models/fasterrcnn_resnet50_fpn_coco-258fb6c6.pth.

A total of 533 bounding boxes with a confidence score of over 70% were predicted across 35 different MS COCO object categories. The aggregate performance can be seen in 4.1. Prediction performance for each object category can be seen in 4.1. Some of the object categories that Faster R-CNN predicted best were: bowl, microwave, and person.

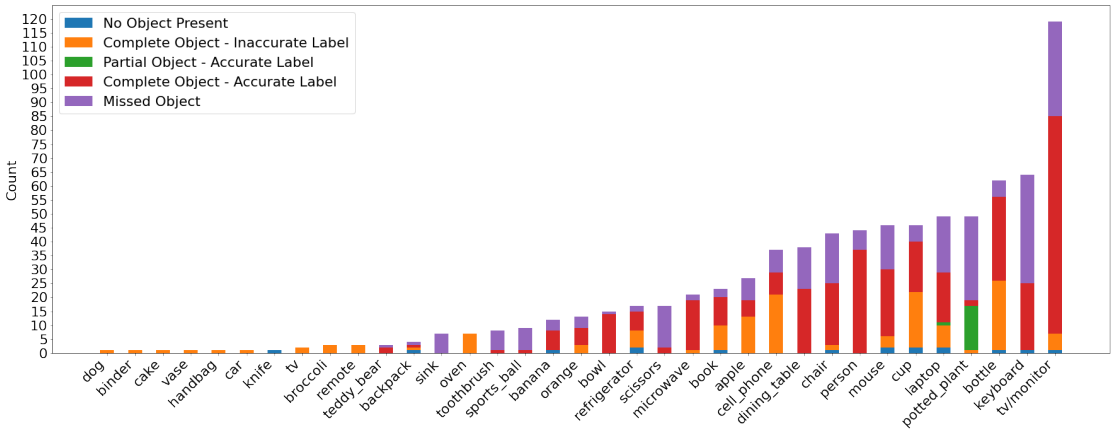


Figure 4.1: Faster R-CNN object detection performance for MS COCO object categories on a subset of ARID. As bounding boxes were not present for all MS COCO object categories, these metrics are the result of manually inspecting predicted bounding boxes.

4.1.1.3 DETR: Detection Transformer

DETR [35] is the first transformer [70] based object detection network that has performance comparable to state of the art methods (such as Faster R-CNN). The network consists of a CNN backbone for feature extraction, a transformer encoder, a transformer decoder, and a feed forward network. The authors claim that the transformer based nature of the network allow it to process the image as a whole using contextual information compared to traditional object detection methods that can only focus on different regions of an image independently.

The DETR [35] implementation offered by Facebook Research available at https://github.com/facebookresearch/detr was used to produce bounding box predictions. The DETR model with a ResNet 50 [66] backbone was used. The publicly available MS COCO pretrained weights can be found at https://dl.fbaipublicfiles.com/detr/detr-r50-e632da11.pth.

Figure 4.2: Two interesting examples of DETR predictions. The left image contains an accurate prediction of a keyboard that is hard to distinguish and that none of the other approaches detected. This seems to give credence to the idea that DETR can successfully use contextual information as the authors claim. The right image contains multiple missed predictions classifying one box of food as multiple books. Images are from [2] and used with the permission of their creator.

A total of 660 bounding boxes with a confidence score of over 70% were predicted across 33 different MS COCO object categories. The aggregate prediction statistics can be seen in 4.1. Out of all of the object detection approaches, DETR had the highest rate of completely accurate predictions as well as the lowest rate of missed object predictions. This was balanced by the fact that it also predicted the most bounding boxes with no object present. Cherry-picked and a lemon-picked predictions can be seen in 4.2. A breakdown of predictions by MS COCO object category can be seen in 4.3. Some of the object categories that DETR performed best with were: bowl, person, and keyboard.

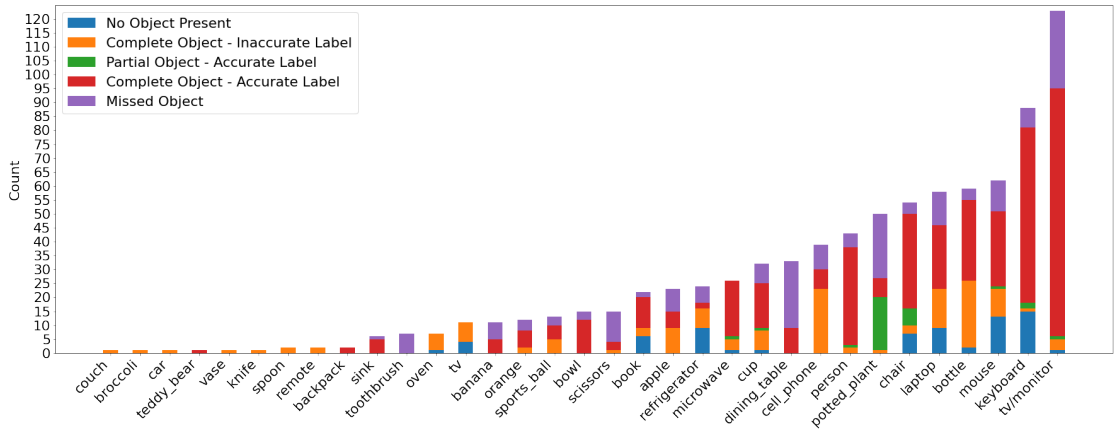


Figure 4.3: DETR object detection performance for MS COCO object categories on a subset of ARID. As bounding boxes were not present for all MS COCO object categories, these metrics are the result of manually inspecting predicted bounding boxes.

4.1.1.4 YOLOv3

YoloV3 [27] is a single stage object detector that builds on YoloV2 [26] by introducing a new deeper feature extractor (Darknet-53) and introduces bounding box predictions over multiple scales. As it is a single stage object detector, it has fast inference times [36].

The YoloV3 implementation offered by Ultralytics available at https://github.com/ultralytics/yolov3 was used to produce bounding box predictions. The YoloV3 MS COCO pretrained weights are available at https://pjreddie.com/media/files/yolov3.weights.

A total of 265 bounding boxes with a confidence score of over 70% were predicted across 26 different MS COCO object categories. The aggregate prediction statistics can be seen in 4.1. YoloV3 performed the worst out of the 3 state of the art deep learning object detection approaches as it had the highest miss rate by a

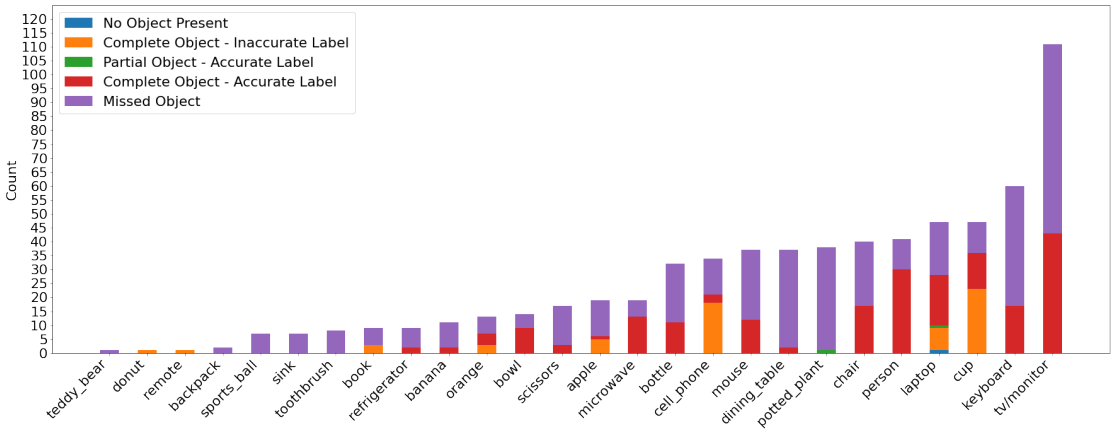


Figure 4.4: YOLO v3 object detection performance for MS COCO object categories on a subset of ARID. As bounding boxes were not present for all MS COCO object categories, these metrics are the result of manually inspecting predicted bounding boxes.

significant margin. It especially struggled with larger items such as the potted plant and dining table. A breakdown of predictions for the different MS COCO object categories can be seen in 4.4.

4.1.2 Commercial Services

4.1.2.1 Google Vision

Google Vision is a commercial service available from Google. It provides a wide range of services such as OCR, face detection, text detection, and object detection. In this work I used the object detection service.

The Google Vision Python API was used to interact with the object detection service. Documentation for the python api can be found at `https://cloud.google.com/vision/docs/object-localizer`. The first 1000 object detection requests were free and it costs $2.25 for every 1000 object detection requests after

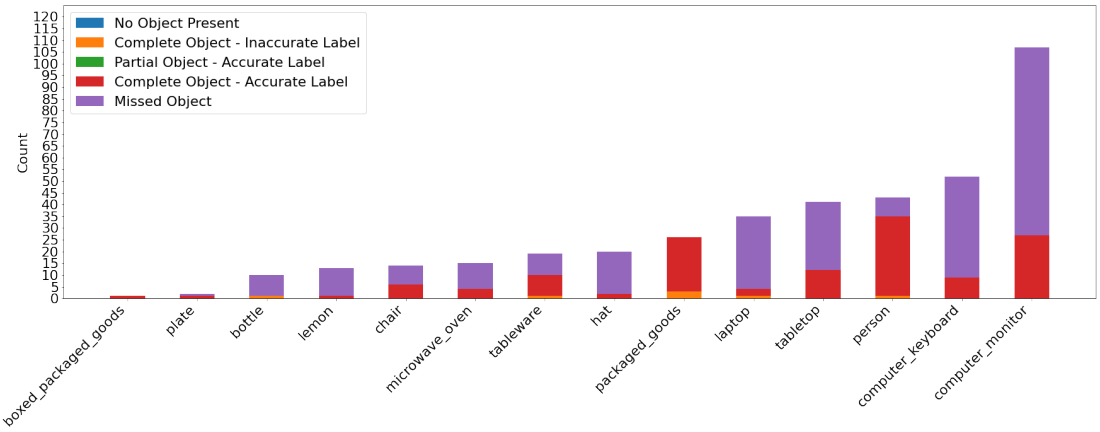


Figure 4.5: Google Vision predictions for each of the different object categories. The predicted bounding boxes were manually evaluated for accuracy.

that.

A total of 139 bounding boxes with a confidence score of over 70% were predicted across 14 different object categories. The aggregate prediction statistics can be seen in 4.1. A breakdown of the predictions of each of the object categories can be seen in 4.5. The Google Vision service performed best at detecting people. The packaged food category did not have any missed items as the category was overly broad and it was hard to accurately identify what constituted a packaged food.

4.1.2.2 Microsoft Azure

The Microsoft Azure Cognitive Service is a set of commercially available artificial intelligence services. It offers services such as face detection, content moderation, speaker detection, and object detection. The object detection service was used here.

The Microsoft Azure Cognitive Service computer vision python api was used to interact with the service. Documentation for the api can be found at `https://docs.`

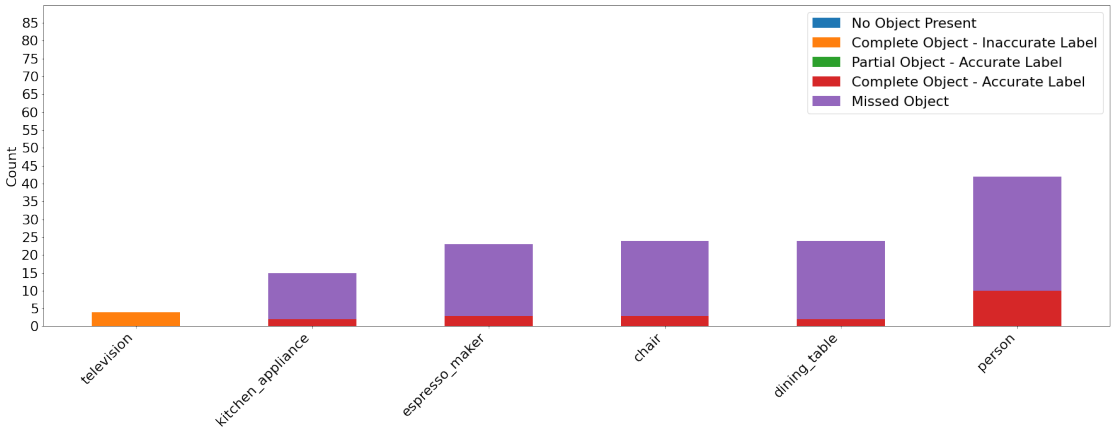


Figure 4.6: Microsoft Azure predictions for each of the different object categories. The predicted bounding boxes were manually evaluated.

microsoft.com/en-us/azure/cognitive-services/computer-vision/quickstarts-sdk/client-library?pivots=programming-language-python. The service cost $1.50 per 1000 image requests.

A total of 24 bounding boxes with a confidence score of over 70% were predicted across 6 different object categories. The aggregate prediction statistics can be seen in 4.1. A breakdown of the predictions of each of the object categories can be seen in 4.6.

4.1.3 Analysis

In this section I have shown the performance of three types of state of the art deep learning based object detection approaches, without training/finetuning on the target dataset, and two commercially available object detection approaches on sixty scenes from ARID, a robotic vision dataset. The deep learning based approaches performed significantly better than the commercial services. DETR, a transformer

Name	Faster R-CNN	DETR	YOLO v3	Google Vision	MS Azure
# of Predictions	533	660	265	139	24
Complete Object Accurate Label	45.2%	49.3%	30.2%	33.2%	15.2%
Partial Object Accurate Label	2.1%	3.8%	0.2%	0.0%	0.0%
Complete Object Inaccurate Label	17.7%	16.8%	9.4%	1.8%	3.0%
No Object	2.0%	8.2%	0.2%	0.0%	0.0%
Missed	33.0%	21.9%	60.0%	65.1%*	81.8%

Table 4.1: Aggregate results for each object detection approach. This data is the result of manually evaluating the bounding box predictions of each object detector on the chosen subset of ARID.

based approach, and Faster R-CNN, a two stage approach, performed comparable to each other and both performed better than YoloV3, a single stage approach. It is interesting to note that all three state of the art deep learning approaches had a significant percentage of predictions that completely bounded an object, but had an inaccurate label, implying that all three of these types of approaches have some lower level notion of "objectness". Looking at the commercially available services, the Google Vision service performed better than the Microsoft Azure service. The Google Vision service both covered more object categories and did so with a higher complete object accuracy than the Microsoft Azure service. Both of the commercially available services tended to be more conservative than the state of the art deep-learning approaches as evidenced by them having a very small percentage of predictions with inaccurate labels, no partial predictions, and no predictions with no object.

4.2 Selective Search Performance

In this section I present the results of running selective search [1], a classical computer vision region proposal approach, on ARID. Selective search is a superpixel based object region proposal approach that identifies possible object locations by combining smaller similar regions together. Selective search was chosen as it has been shown to perform reasonably well over another robotic vision dataset [19] as well as being the region proposal method used in R-CNN [30]. Three separate variants of selective search: *single*, *fast*, and *quality* were used on the entirety of ARID. The implementation used to generate the region proposals can be found at https://github.com/ChenjieXu/selective_search. The intersection over union metric (IOU) was used to evaluate the performance of each selective search variant. The IOU metric is a standard metric for measuring the region proposal portion of object detection approaches [4–6, 71]. Three separate IOU values were considered: 0.5, 0.7, and 0.9.

4.2.1 Analysis

As can be seen in 4.2, the *quality* variant of selective search performed the best, but at the cost of increased execution time as well as drastically more region proposals that would need to be processed by a classifier. Across the three variants, selective search struggled with a similar set of objects (rubber_eraser, dry_battery, etc.) that I believe is due to the small object size and/or varied coloring. On the other hand, all selective search variants were more successful with a similar set

of objects (bell_pepper, keyboard, hand_towel, etc.) that I believe is due to their uniform color and/or larger size. Overall, the results from this experiment show that selective search can be a useful tool for generating region proposals for object detection in robotic vision environments.

Variant	Avg. # of Proposals	Avg. Exec. Time	0.5 IOU	0.7 IOU	0.9 IOU
Single	770	12 seconds	78%	45%	4.5%
Fast	3397	24 seconds	91%	67%	8.8%
Quality	17952	143 seconds	99%	92%	37%

Table 4.2: Aggregate results for each selective search variant. The average proposal count and average execution time are per image. The IOU columns represent the total object instance recall at those IOU thresholds.

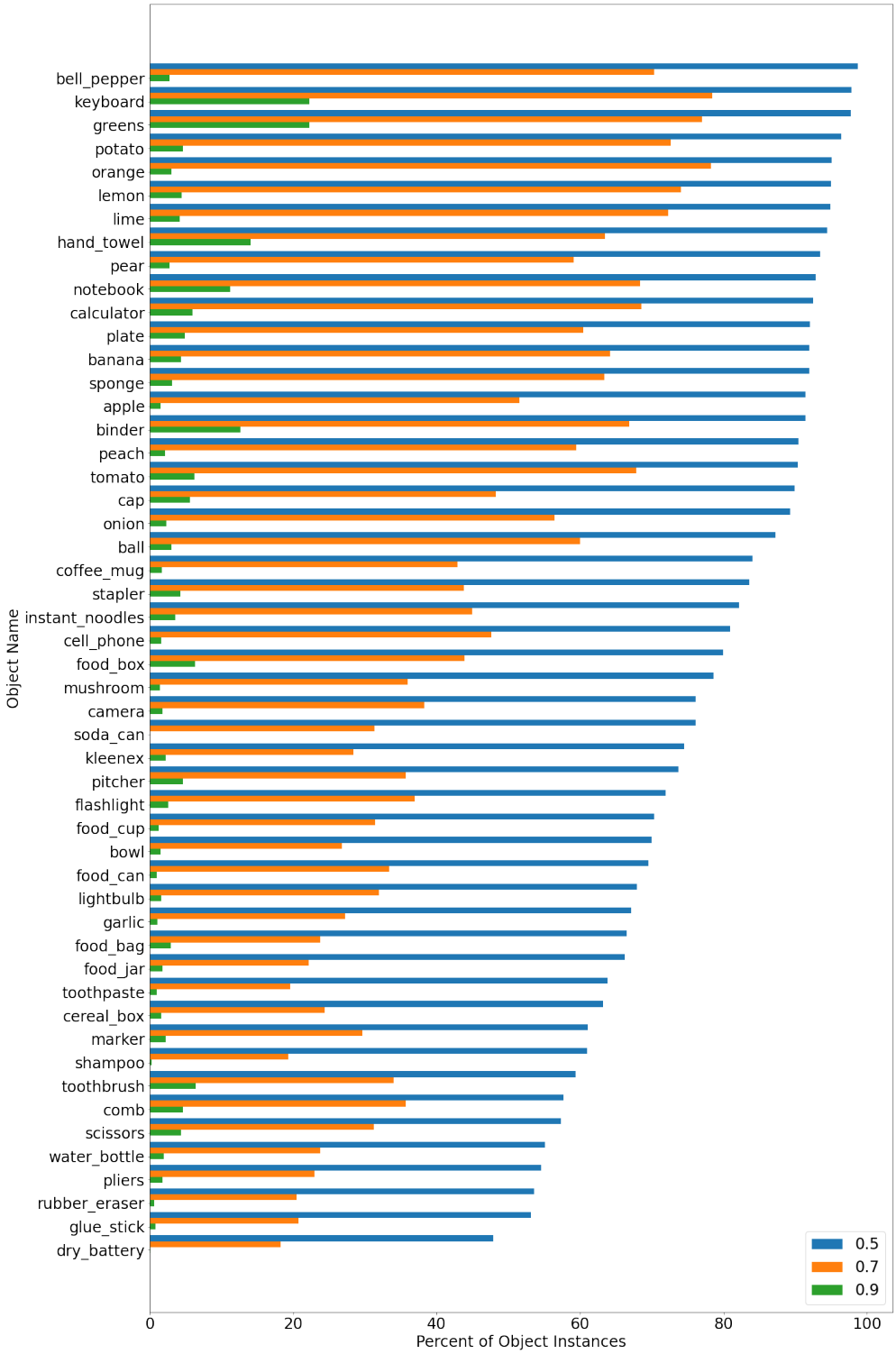


Figure 4.7: Percent of ARID object instance bounding boxes that were discovered by generated regions at three IOU score thresholds for the selective search *single* variant.

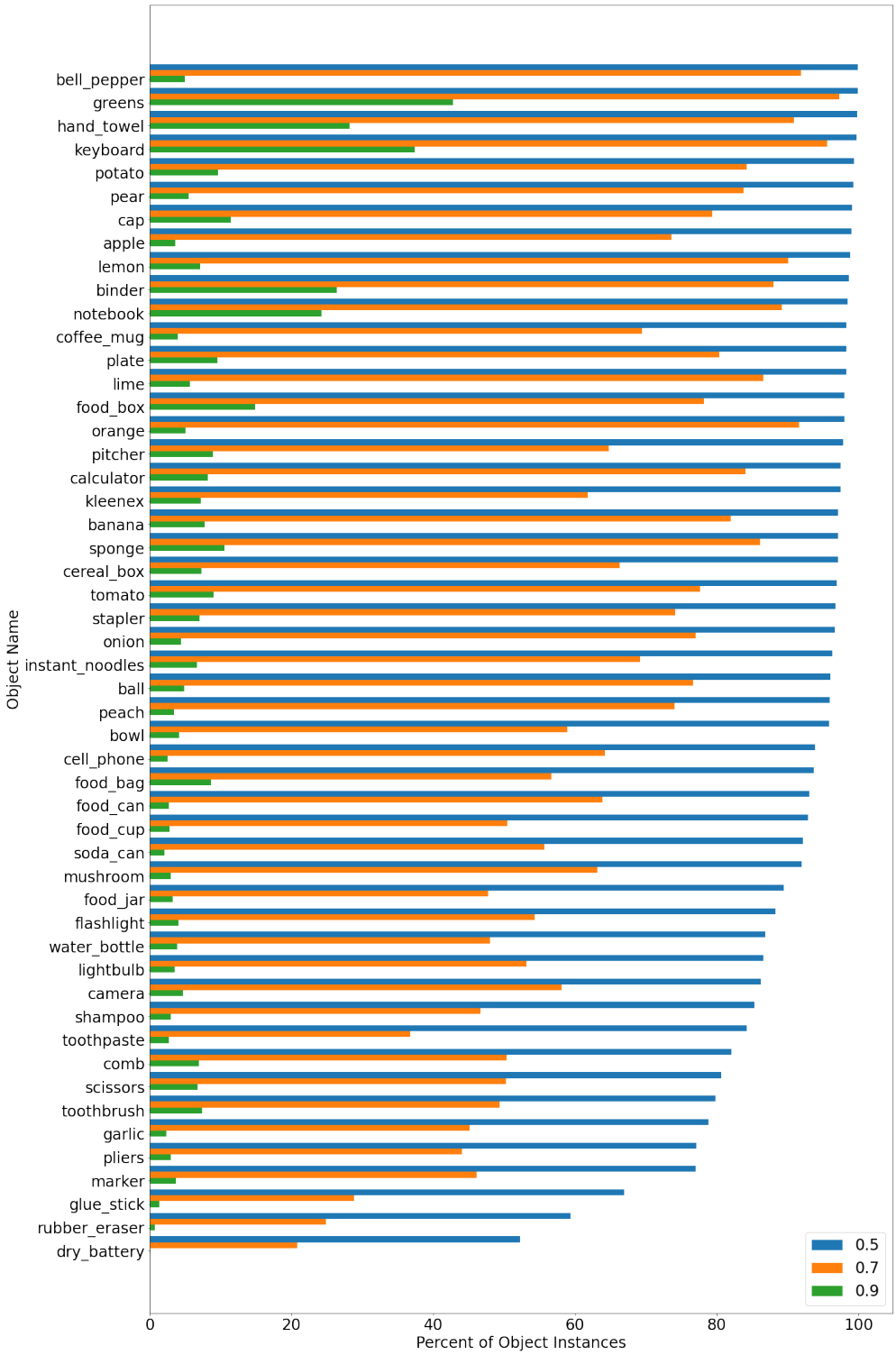


Figure 4.8: Percent of ARID object instance bounding boxes that were discovered by generated regions at three IOU score thresholds for the selective search *fast* variant.

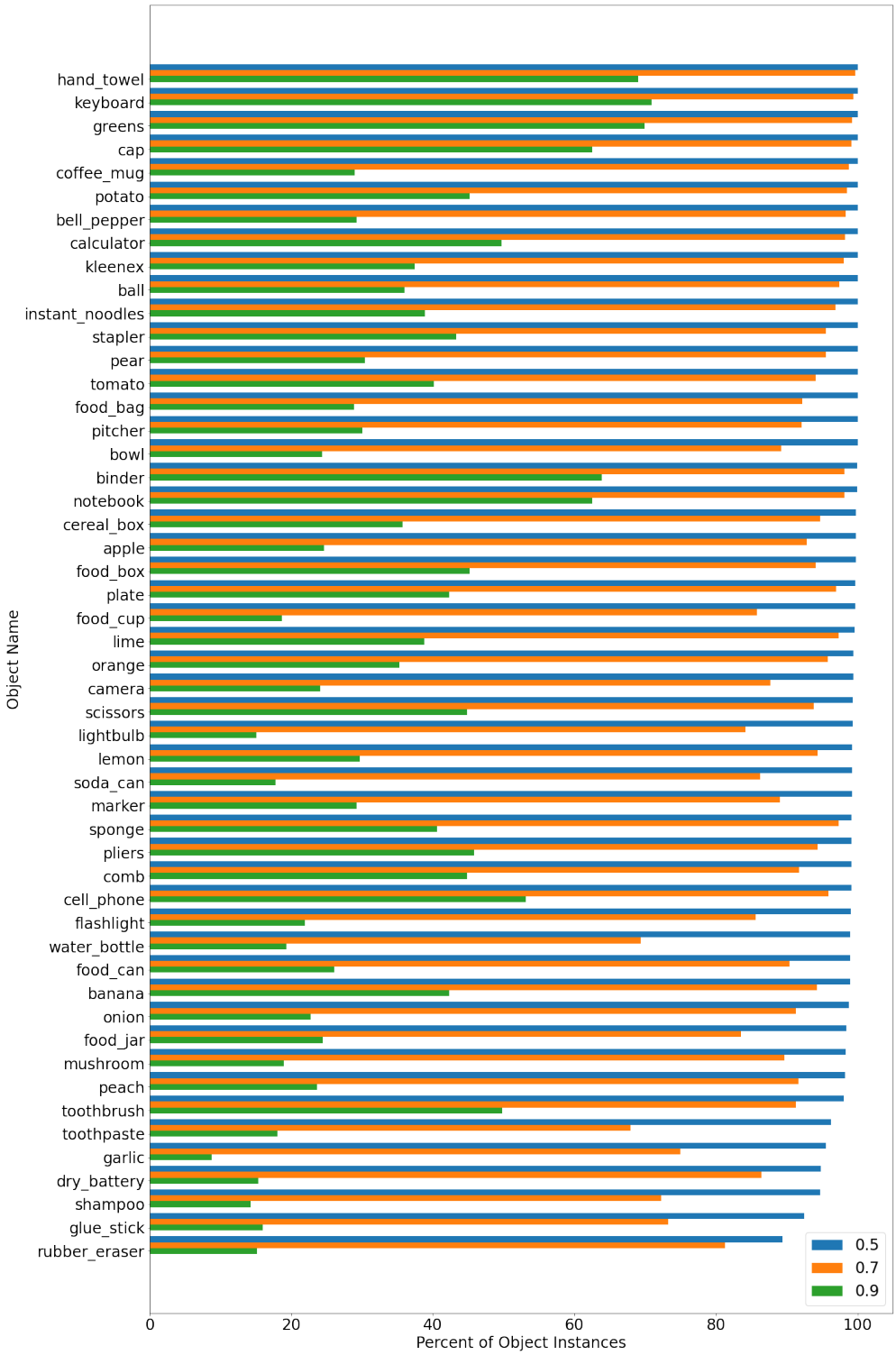


Figure 4.9: Percent of object instance bounding boxes that were discovered by generated regions at three IOU score thresholds for the selective search *quality* variant.

4.3 Evaluating the Binary Classifiers on ARID Crops

In this section I evaluate the performance of the introduced approach's webly supervised binary classifiers on the ARID object instance crops. The goal of this experiment is to gain an understanding as to how well the binary classifiers' learned representations maps to the associated ARID object instance. For each trained classifier, the ARID object instance used to generate the description for its webly dataset is considered the positive class, while all object instances from outside the target object instance's category is considered the negative class (e.g. apple_1 and apple_2 crops would not be included in either the positive or negative class when evaluating the apple_0 classifier, but lemon_0 crops would be). The precision-recall curve is used as the evaluation metric as it allows us to see the classifiers' performances over multiple thresholds which is beneficial as the system does not require an absolute classification threshold, but a relative one as it uses the top scoring region proposal as its prediction.

4.3.1 Analysis

Considering the precision-recall curves for each of the binary classifiers 4.10 there are two indicators that a binary classifier will perform well. The strongest indicator is that the target object instance has a unique color/shape compared to the other object instances (e.g. ball_0, bell_pepper_1, and coffee_mug_2). Another performance indicator is how visually near the associated webly dataset is to the actual ARID object instance crops (e.g. orange_*, tomato_0, and cereal_box_0). It is

important to note that as the negative class used in training the classifiers consisted of MS COCO object categories and that some MS COCO categories overlap with ARID object categories, it is possible that this had a negative impact on some classifiers.

apple_0	apple_1	apple_2	ball_0	banana_0
0.344	0.121	0.408	0.884	0.4

banana_1	banana_2	bell_pepper_0	bell_pepper_1	bell_pepper_2
0.34	0.359	0.221	0.865	0.495

binder_0	bowl_0	bowl_1	bowl_2	calculator_0
0.044	0.284	0.512	0.455	0.294

camera_0	cap_2	cell_phone_0	cell_phone_1	cell_phone_2
0.226	0.187	0.281	0.382	0.099

cereal_box_0	coffee_mug_0	coffee_mug_1	coffee_mug_2	comb_0
0.516	0.439	0.238	0.826	0.189

dry_battery_0	flashlight_2	greens_0	keyboard_0	keyboard_1
0.132	0.043	0.307	0.177	0.454

keyboard_2	lemon_0	lightbulb_0	mushroom_0	orange_0
0.378	0.384	0.093	0.079	0.379

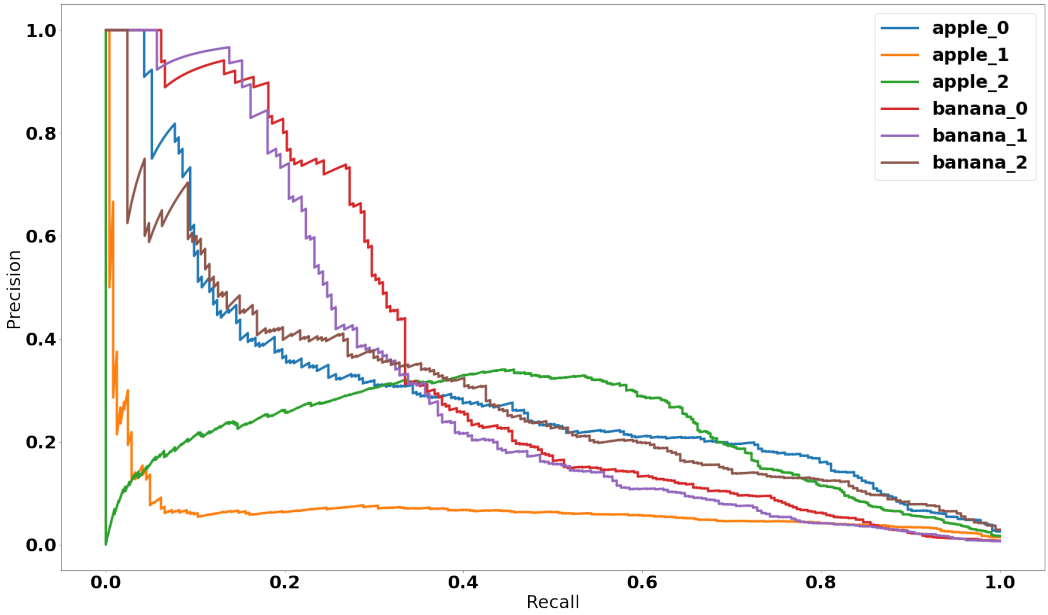
orange_1	orange_2	pliers_1	potato_0	scissors_0
0.3	0.345	0.063	0.144	0.275

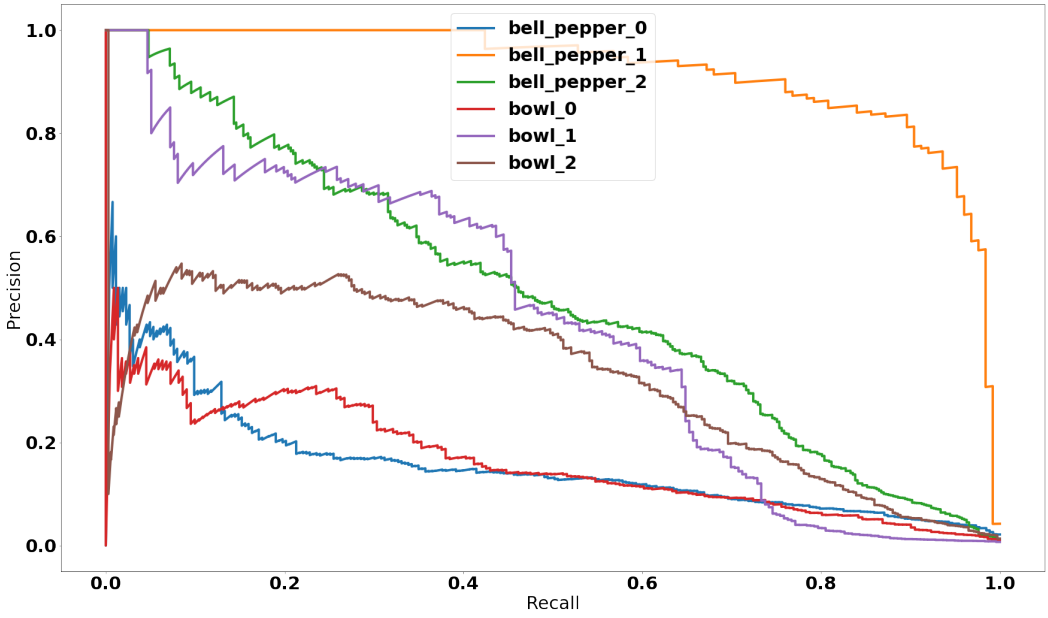
scissors_1	scissors_2	shampoo_0	sponge_0	stapler_0
0.072	0.162	0.039	0.157	0.166

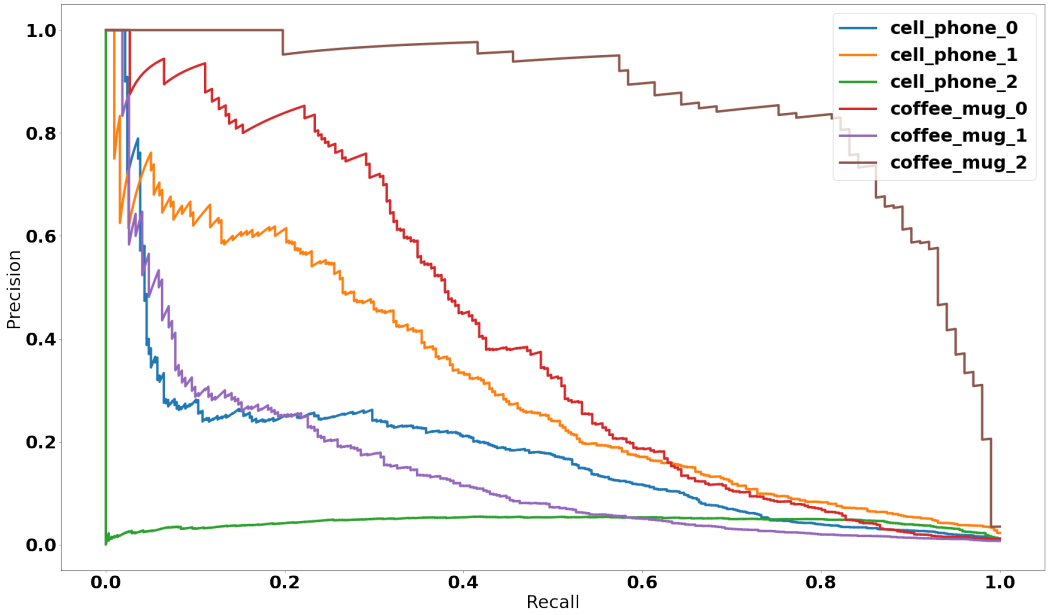
tomato_0	toothbrush_0	toothbrush_2	toothpaste_0	water_bottle_0
0.32	0.073	0.094	0.052	0.012

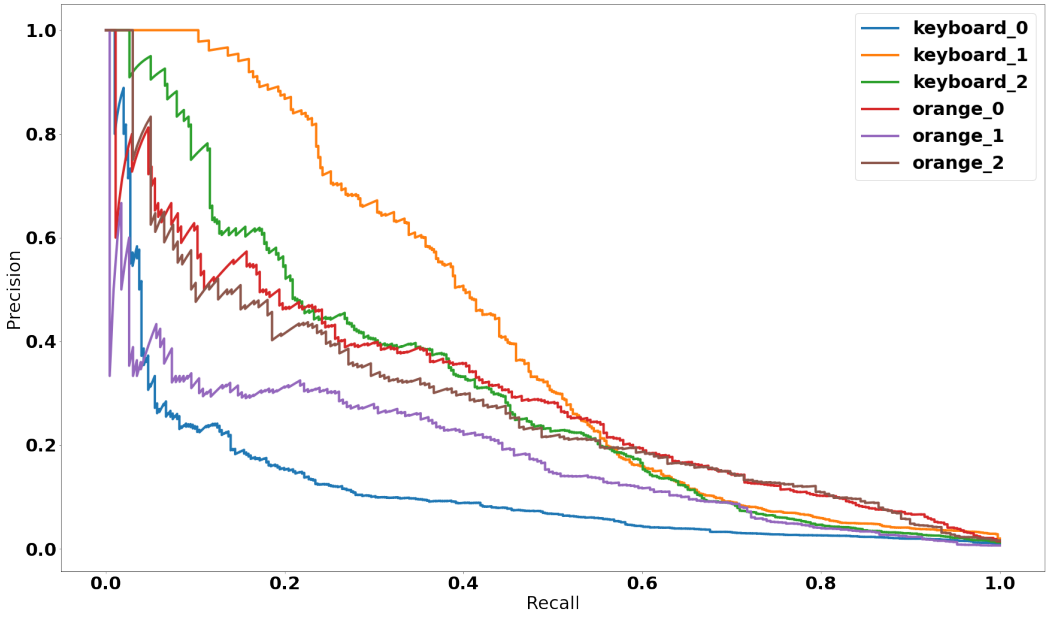
water_bottle_1	water_bottle_2
0.007	0.118

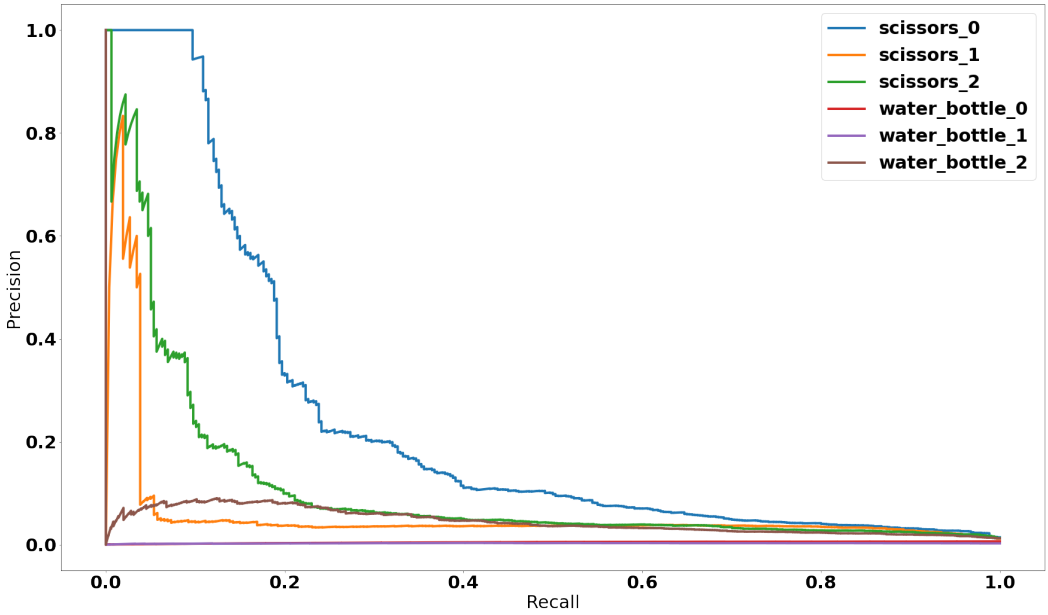
Table 4.3: F1 scores for each object instances' binary classifier when evaluating on ARID crops.

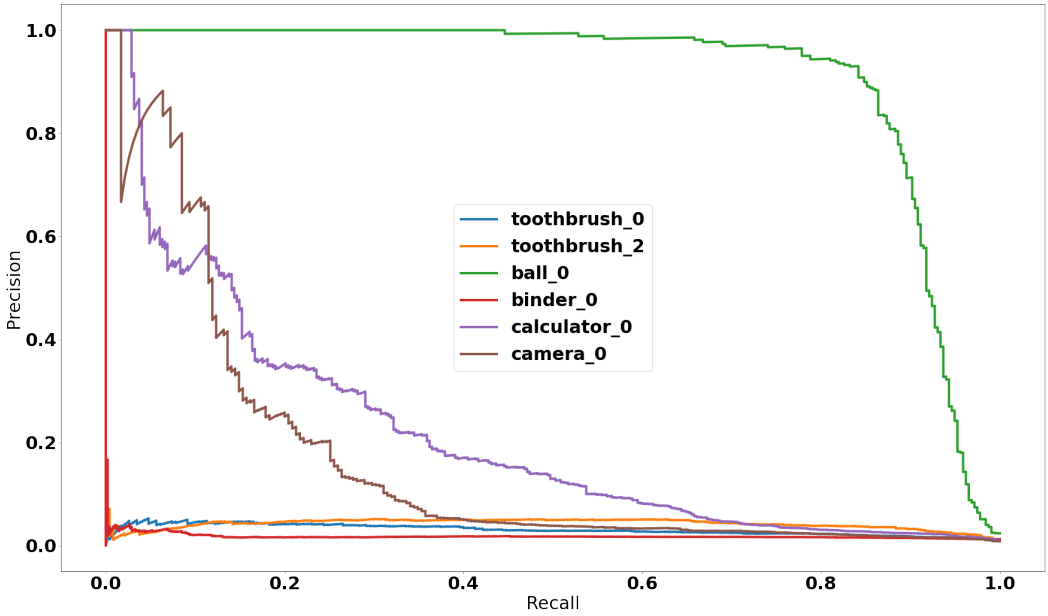

apple_0
apple_1
apple_2
banana_0
banana_1
banana_2
Precision
Recall
0.0
0.2
0.4
0.6
0.8
1.0

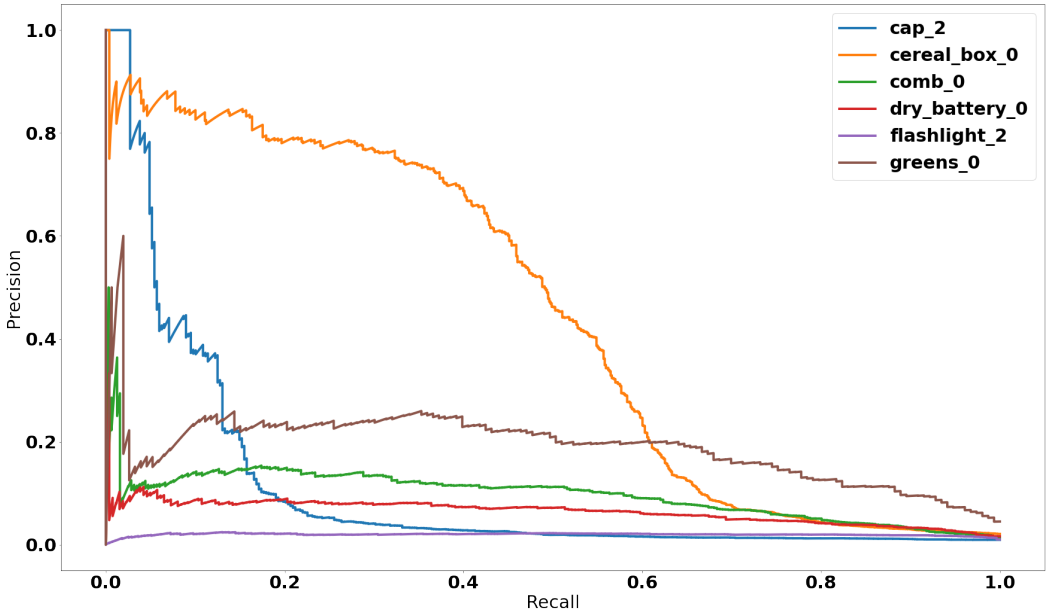

bell_pepper_0
bell_pepper_1
bell_pepper_2
bowl_0
bowl_1
bowl_2
Precision
Recall
0.0
0.2
0.4
0.6
0.8
1.0

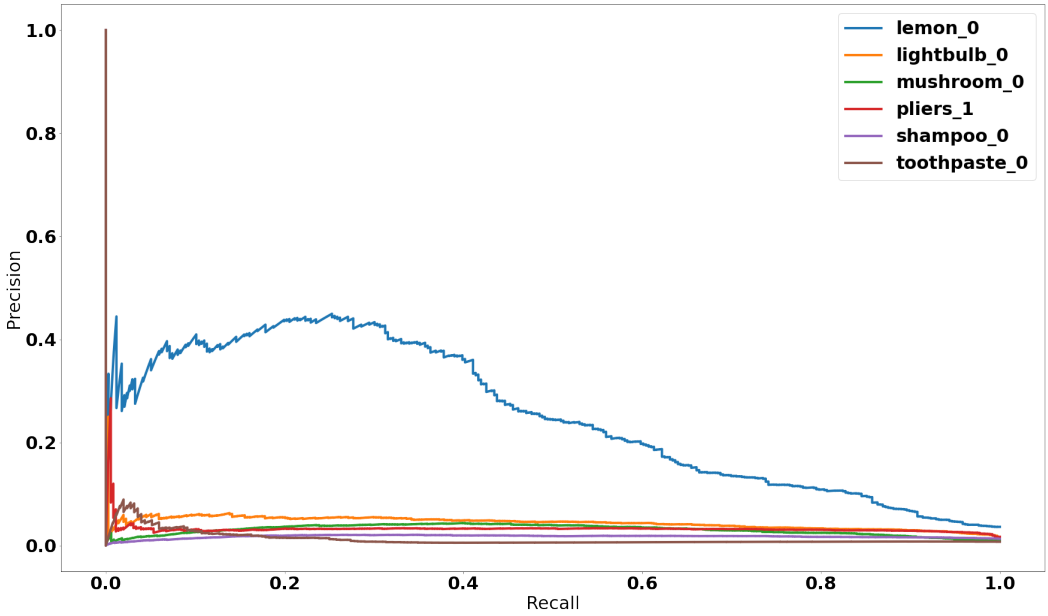

cell_phone_0
cell_phone_1
cell_phone_2
coffee_mug_0
coffee_mug_1
coffee_mug_2
Precision
Recall
0.0
0.2
0.4
0.6
0.8
1.0


keyboard_0
keyboard_1
keyboard_2
orange_0
orange_1
orange_2
Precision
Recall
0.0
0.2
0.4
0.6
0.8
1.0


scissors_0
scissors_1
scissors_2
water_bottle_0
water_bottle_1
water_bottle_2
Precision
Recall
1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0


toothbrush_0
toothbrush_2
ball_0
binder_0
calculator_0
camera_0
Precision
Recall
1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0


cap_2
cereal_box_0
comb_0
dry_battery_0
flashlight_2
greens_0
Precision
Recall
0.0
0.2
0.4
0.6
0.8
1.0


lemon_0
lightbulb_0
mushroom_0
pliers_1
shampoo_0
toothpaste_0
Precision
Recall
0.0
0.2
0.4
0.6
0.8
1.0

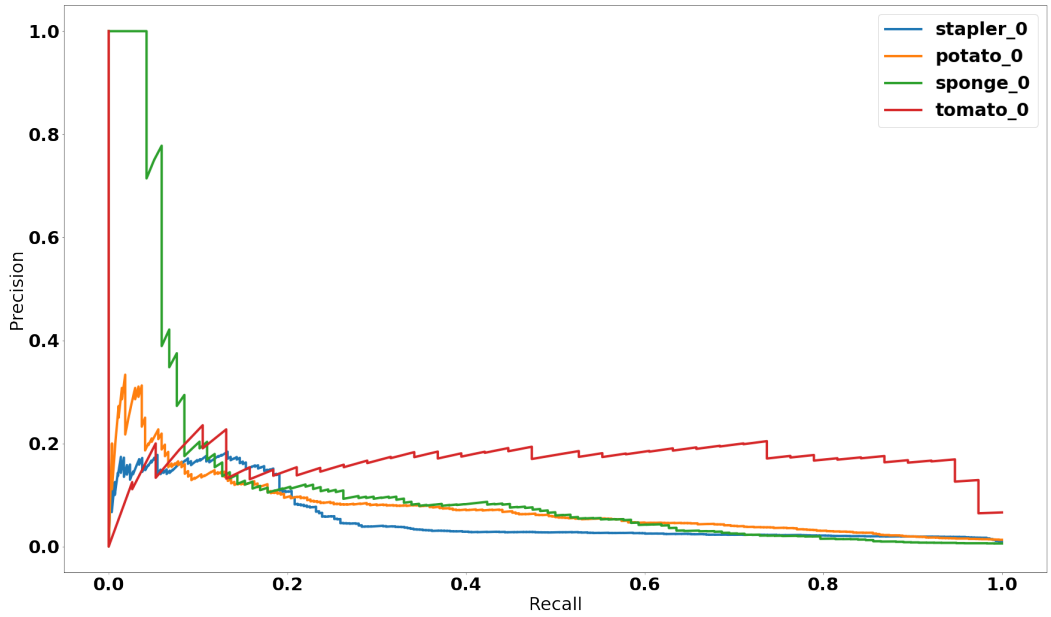


Figure 4.10: Precision-recall curves for the per object instance trained binary classifiers when evaluated on the ARID crops. Other object instances from the same object category as the target object were excluded from the evaluation due to possible overlap between object instance descriptions (e.g. lemon_1 and lemon_2 were excluded when evaluating lemon_0).

4.4 End to End Evaluation of the Introduced Approach

In this section I evaluate the end to end performance of the introduced approach. I specifically consider the intersection over union (IOU) metric as it is a commonly used metric for object detection [4–6, 71]. Two IOU metric thresholds were considered: 0.1 and 0.5. 0.5 is a standard baseline for object detection while 0.1 indicates an approach's ability to locate some discriminative region of an object in an image. As the approach introduced in this work ranks object region proposals without modifying them and without training on ground truth bounding boxes, the most discriminative regions tend to be scored the highest by the webly supervised binary classifier. The top one and top five predictions were considered. The top one prediction was the highest scoring region proposal. The top five predictions were found by applying non-maximum suppression with an IOU of 0.1 and then the top five scoring region proposals were selected as predictions.

4.4.1 Analysis

The performance of the top one and the top five predictions for each object instance can be seen in 4.11 and 4.12. There was a large performance gap between predictions that met an IOU threshold of 0.1 and 0.5 which is a testament to the need for some kind of region proposal refinement. There was not a large performance gap between the top one and top five predictions. The gap between top one and top five prediction performance may change with a region proposal refinement approach and a better way to handle region proposals that identify the same object, but do

not overlap which are not affected by non-maximum suppression.

A good performance indicator for how well the approach is able to localize an object is how well its associated binary classifier performed on the ARID crops, however this was not always the case (e.g. tomato_0). Two possible explanations for this are that object instances were routinely located in scenes with few other objects near in color or shape which would make localizing the object easier or that the region proposals generated by selective search isolated more discriminative regions of the object allowing for a better classification score than the entire object would receive.

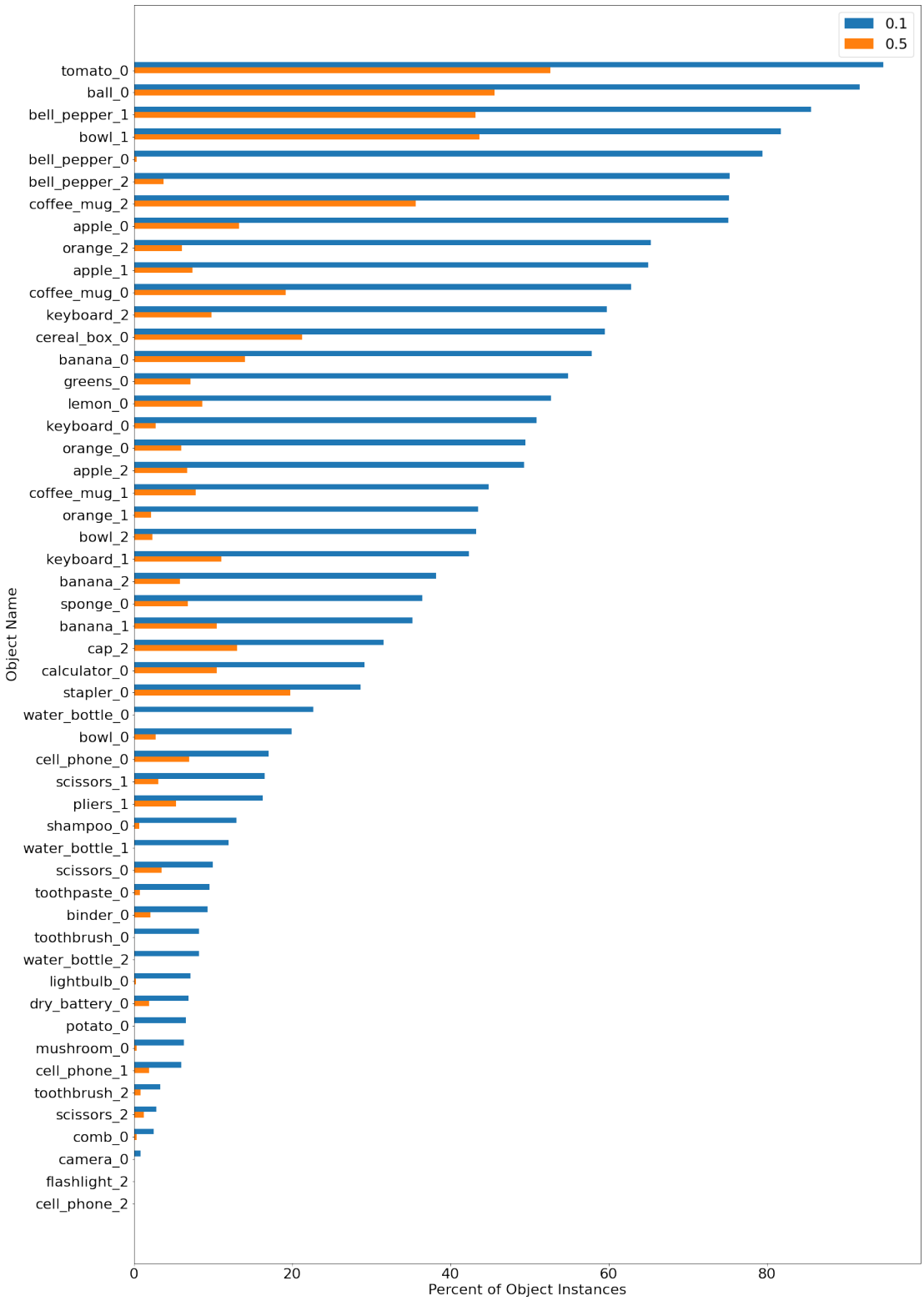


Figure 4.11: Percent of object instance bounding boxes covered by the top predicted bounding box at two IOU thresholds.

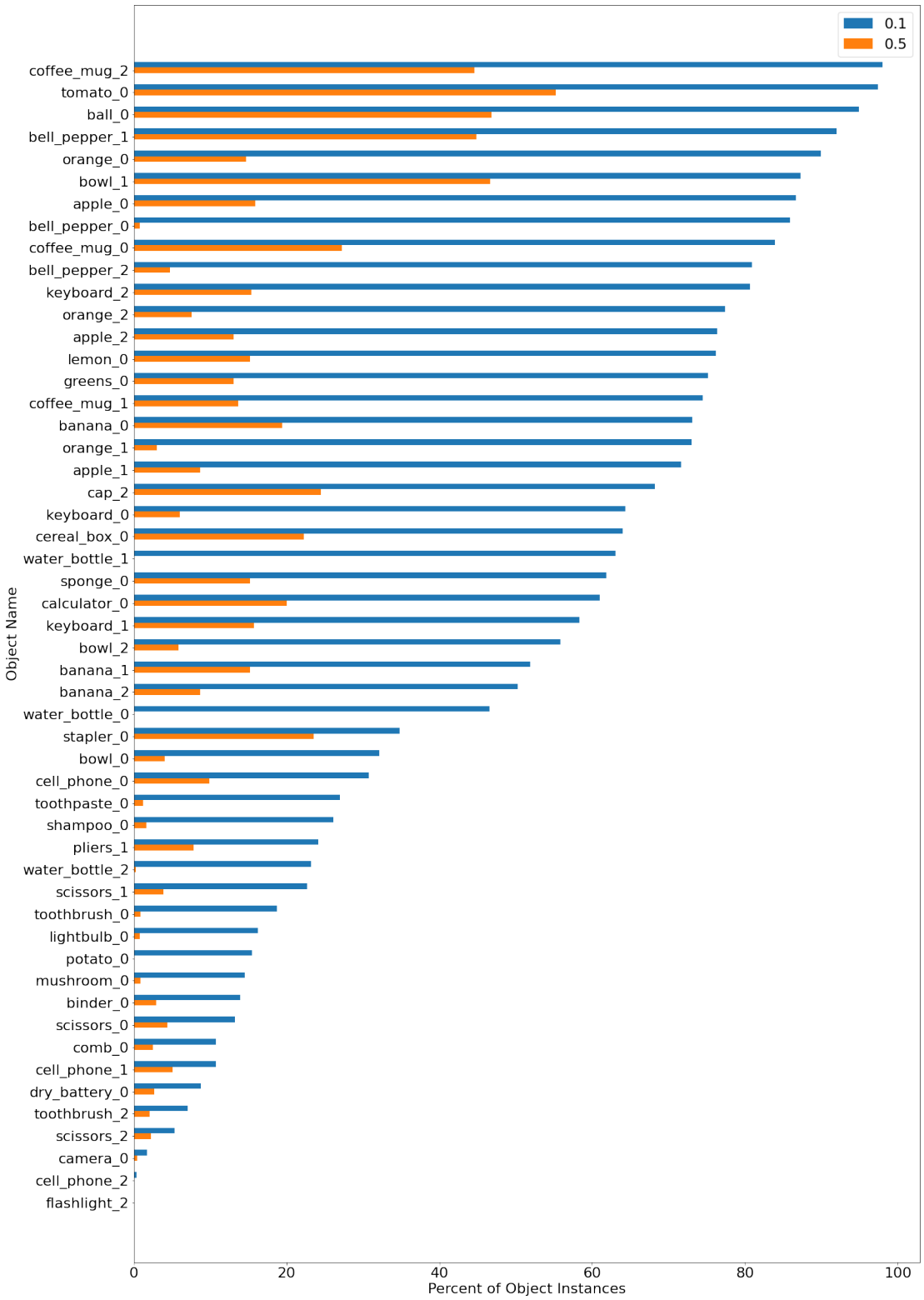


Figure 4.12: Percent of object instance bounding boxes covered by at least one of the top five predicted bounding boxes at two IOU thresholds.

4.5 Comparing to Faster R-CNN

In this section I compare the performance of the introduced approach to Faster R-CNN [33] at localizing object categories present in both ARID and MS COCO. Faster R-CNN was chosen as it was one of the highest performing object detectors evaluated in 4.1 and it is also a multiple stage approach like the approach introduced in this work. Faster R-CNN predictions were chosen by selecting the predicted bounding box with the highest confidence score for each object category present. The evaluation metric used was IOU with a threshold of 0.1 as the introduced approach struggled with values greater than that as noted in 4.4.

4.5.1 Analysis

The introduced approach's and Faster R-CNN's performance at localizing each object instance from all object categories shared between MS COCO and ARID can be seen in 4.13. Generally the approach introduced in this paper performed worse than Faster R-CNN. For the object categories *apple*, *orange*, *keyboard*, and *coffee_mug* performance was comparable between the two approaches. I believe this is due to the solid colors of those objects and generally homogeneous shapes of them. The introduced approach did end up outperforming Faster R-CNN when trying to localize *apple_1* which is a yellow apple. I believe that the non-standard color of the apple led to Faster R-CNN struggling to localize it, while the introduced approach was able to leverage the descriptive information it was given to localize it.

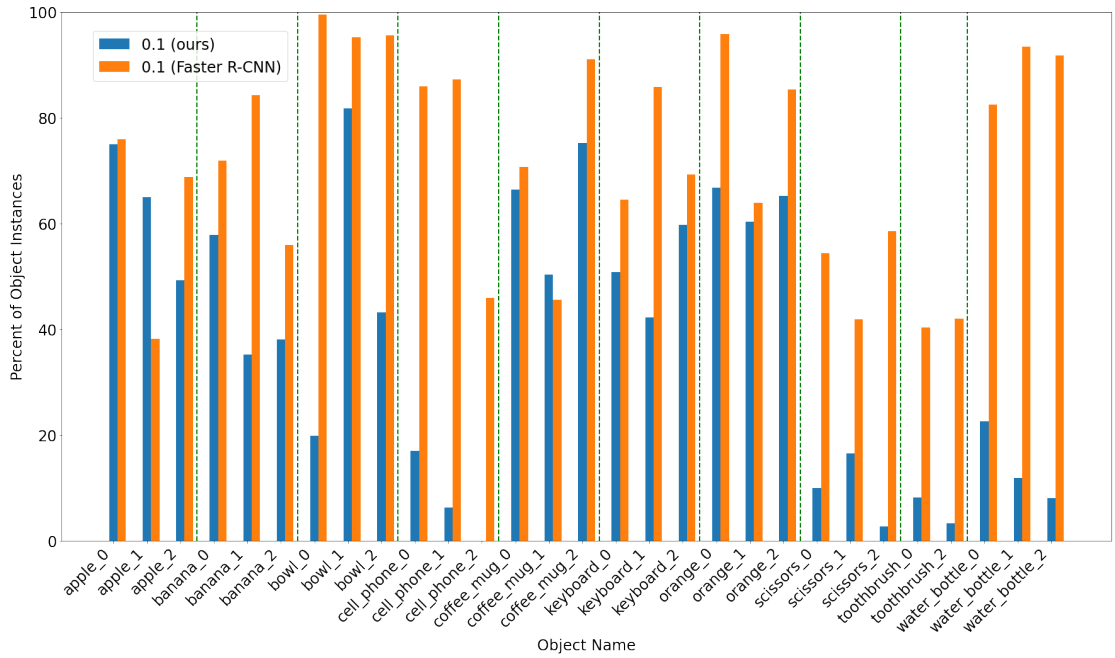


Figure 4.13: Percent of object instances from object categories present in both MS COCO and ARID that were bounded at an IOU threshold of 0.1 for the approach introduced in this paper and Faster R-CNN.

Chapter 5: Conclusion and Future Work

5.1 Conclusion

In this work I introduced a three stage natural language guided webly supervised object localization approach that could be used in a robotics environment to detect novel objects. I first established a performance baseline by evaluating the performance of three state of the art deep learning based object detection approaches and two commercial object detection services on a robotic vision dataset. I then conducted experiments to evaluate the performance of the separate stages of the introduced approach as well as the end to end performance of the approach. Finally I compared the performance of the introduced approach to one of the top performing state of the art object detection approaches. In summary, the main contributions of this work are as follows:

- Deep learning based object detection approaches can perform well on robotic vision datasets without being trained specifically for this task.

- The quality of webly datasets generated from natural language descriptions of objects vary widely. Webly datasets created from objects that are generally homogeneous in their appearance and are mono-colored tend to be of a much

higher quality than webly datasets generated from more complex objects.

- The approach introduced in this paper generally performs worse at localizing objects in a robotics dataset than one of the top deep learning based approaches on known object categories, but can perform at a comparable level when the target object meets the criteria needed for the creation of a high quality webly dataset.

5.2 Future Work

I believe that there are many ways to build upon the work introduced in this paper. Looking at each individual stage of the introduced approach, there are many areas for improvement. First, applying some kind of smarter domain adaptions to the webly datasets to better align them to the real world objects that they are trying to localize would improve performance as shown in 4.3 and 4.4. Next, replacing the binary classifier with a single class classifier (such as the approach introduced in [72]) would ensure that a target object was never present in the negative class which could improve confidence scores. Finally, replacing selective search with a smarter region proposal system, such as a region proposal network that could detect class-agnostic objects, could both improve localization as well as decrease the number of proposals that need to be processed. Another improvement to the region proposal system would be to find a way to combine multiple proposals that were located near each other into a singular proposal. Other interesting areas for further experimentation are applying the system to other datasets and gathering object descriptions from

multiple sources to investigate how object descriptions impact performance.

Bibliography

[1] J. R. R. Uijlings, K. E. A. van de Sande, T. Gevers, and A. W. M. Smeulders. Selective search for object recognition. International Journal of Computer Vision, 104(2):154–171, 2013.

[2] Mohammad Reza Loghmani, Barbara Caputo, and Markus Vincze. Recognizing objects in-the-wild: Where do we stand? In IEEE International Conference on Robotics and Automation (ICRA), 2018.

[3] Cynthia Matuszek. Grounded language learning: Where robotics and nlp meet. In IJCAI, 2018.

[4] M. Everingham, S. M. A. Eslami, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The pascal visual object classes challenge: A retrospective. International Journal of Computer Vision, 111(1):98–136, January 2015.

[5] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. International Journal of Computer Vision (IJCV), 115(3):211–252, 2015.

[6] Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, C. Lawrence Zitnick, and Piotr Dollár. Microsoft coco: Common objects in context, 2014.

[7] Xinlei Chen and Abhinav Gupta. Webly supervised learning of convolutional networks. In Proceedings of the IEEE International Conference on Computer Vision (ICCV), December 2015.

[8] Santosh K. Divvala, Ali Farhadi, and Carlos Guestrin. Learning everything about anything: Webly-supervised visual concept learning. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2014.

[9] Bin Jin, Maria V. Ortiz Segovia, and Sabine Susstrunk. Webly supervised semantic segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), July 2017.

[10] Alessandro Bergamo and Lorenzo Torresani. Exploiting weakly-labeled web images to improve object classification: a domain adaptation approach. In J. D. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R. S. Zemel, and A. Culotta, editors, Advances in Neural Information Processing Systems 23, pages 181–189. Curran Associates, Inc., 2010.

[11] Nizar Massouh, Francesca Babiloni, Tatiana Tommasi, Jay Young, Nick Hawes, and Barbara Caputo. Learning deep visual object models from noisy web data: How to make it work. 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Sep 2017.

[12] Massimilano Mancini, Hakan Karaoguz, Elisa Ricci, Patric Jensfelt, and Barbara Caputo. Knowledge is never enough: Towards web aided deep open world recognition. In IEEE International Conference on Robotics and Automation (ICRA), May 2019.

[13] Qingyi Tao, Hao Yang, and Jianfei Cai. Zero-annotation object detection with web knowledge transfer, 2018.

[14] Li Niu, Qingtao Tang, Ashok Veeraraghavan, and Ashu Sabharwal. Learning from noisy web data with category-level supervision, 2018.

[15] Yi Tu, Li Niu, Junjie Chen, Dawei Cheng, and Liqing Zhang. Learning from web data with self-organizing memory module, 2020.

[16] Junnan Li, Caiming Xiong, and S. Hoi. Mopro: Webly supervised learning with momentum prototypes. ArXiv, abs/2009.07995, 2020.

[17] K. Lai, L. Bo, X. Ren, and D. Fox. A large-scale hierarchical multi-view rgb-d object dataset. In 2011 IEEE International Conference on Robotics and Automation, pages 1817–1824, 2011.

[18] K. Lai, L. Bo, and D. Fox. Unsupervised feature learning for 3d scene labeling. In 2014 IEEE International Conference on Robotics and Automation (ICRA), pages 3050–3057, 2014.

[19] Georgios Georgakis, Md Alimoor Reza, Arsalan Mousavian, Phi-Hung Le, and Jana Kosecka. Multiview rgb-d dataset for object instance detection, 2016.

[20] Qi She, Fan Feng, Xinyue Hao, Qihan Yang, Chuanlin Lan, Vincenzo Lomonaco, Xuesong Shi, Zhengwei Wang, Yao Guo, Yimin Zhang, Fei Qiao, and Rosa H. M. Chan. OpenLORIS-Object: A robotic vision dataset and benchmark for lifelong deep learning. In 2020 International Conference on Robotics and Automation (ICRA), pages 4767–4773, 2020.

[21] Georgios Georgakis, Arsalan Mousavian, Alexander C. Berg, and Jana Kosecka. Synthesizing training data for object detection in indoor scenes, 2017.

[22] L. Jiao, F. Zhang, F. Liu, S. Yang, L. Li, Z. Feng, and R. Qu. A survey of deep learning-based object detection. IEEE Access, 7:128837–128868, 2019.

[23] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C. Berg. Ssd: Single shot multibox detector. Lecture Notes in Computer Science, page 21–37, 2016.

[24] T. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár. Focal loss for dense object detection. In 2017 IEEE International Conference on Computer Vision (ICCV), pages 2999–3007, 2017.

[25] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection, 2016.

[26] Joseph Redmon and Ali Farhadi. Yolo9000: Better, faster, stronger, 2016.

[27] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. arXiv, 2018.

[28] Shifeng Zhang, Longyin Wen, Xiao Bian, Zhen Lei, and Stan Z. Li. Single-shot refinement neural network for object detection. In CVPR, 2018.

[29] Qijie Zhao, Tao Sheng, Yongtao Wang, Zhi Tang, Ying Chen, Ling Cai, and Haibin Ling. M2det: A single-shot object detector based on multi-level feature pyramid network, 2019.

[30] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation, 2014.

[31] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. Lecture Notes in Computer Science, page 346–361, 2014.

[32] Ross Girshick. Fast r-cnn, 2015.

[33] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks, 2015.

[34] Jifeng Dai, Yi Li, Kaiming He, and Jian Sun. R-fcn: Object detection via region-based fully convolutional networks, 2016.

[35] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers, 2020.

[36] Petru Soviany and Radu Tudor Ionescu. Optimizing the trade-off between single-stage and two-stage object detectors using image difficulty prediction, 2018.

[37] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition, 2015.

[38] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection, 2017.

[39] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, Advances in Neural Information Processing Systems 25, pages 1097–1105. Curran Associates, Inc., 2012.

[40] W. Li, L. Wang, Wei Li, Eirikur Agustsson, and L. Gool. Webvision database: Visual learning and understanding from web data. ArXiv, abs/1708.02862, 2017.

[41] Reaz Ashraful Abedin, S. Bensch, and T. Hellström. Self-supervised language grounding by active sensing combined with internet acquired images and text. 2017.

[42] L. Smyth, D. Kangin, and N. Pugeault. Training-valuenet: Data driven label noise cleaning on weakly-supervised web images. In 2019 Joint IEEE 9th International Conference on Development and Learning and Epigenetic Robotics (ICDL-EpiRob), pages 307–312, 2019.

[43] Jingkang Yang, Weirong Chen, Litong Feng, Xiaopeng Yan, Huabin Zheng, and Wayne Zhang. Webly supervised image classification with metadata: Automatic noisy label correction via visual-semantic graph. Proceedings of the 28th ACM International Conference on Multimedia, 2020.

[44] S. Guo, Weilin Huang, H. Zhang, Chenfan Zhuang, Dengke Dong, M. Scott, and Dinglong Huang. Curriculumnet: Weakly supervised learning from large-scale web images. ArXiv, abs/1808.01097, 2018.

[45] Lu Jiang, Zhengyuan Zhou, Thomas Leung, Li-Jia Li, and Li Fei-Fei. Mentornet: Learning data-driven curriculum for very deep neural networks on corrupted labels, 2018.

[46] D. Cheng, F. Setti, N. Zeni, R. Ferrario, and Marco Cristani. Semantically-driven automatic creation of training sets for object recognition. Comput. Vis. Image Underst., 131:56–71, 2015.

[47] Tomas Kulvicius, Irene Markelic, M. Tamosiunaite, and F. Wörgötter. Semantic image search for robotic applications. ArXiv, abs/2004.02607, 2020.

[48] Fernando Navarro, Sailesh Conjeti, Federico Tombari, and Nassir Navab. Webly supervised learning for skin lesion classification. Lecture Notes in Computer Science, page 398–406, 2018.

[49] Dario Molinari, Giulia Pasquale, Lorenzo Natale, and Barbara Caputo. Automatic creation of large scale object databases from web resources: A case study in robot vision. In Elisa Ricci, Samuel Rota Bulò, Cees Snoek, Oswald Lanz, Stefano Messelodi, and Nicu Sebe, editors, Image Analysis and Processing – ICIAP 2019, pages 488–498, Cham, 2019. Springer International Publishing.

[50] Raymond J. Mooney. Learning to connect language and perception. In Proceedings of the 23rd AAAI Conference on Artificial Intelligence (AAAI), pages 1598–1601, Chicago, IL, July 2008. Senior Member Paper.

[51] Stefanie Tellex, Nakul Gopalan, Hadas Kress-Gazit, and Cynthia Matuszek. Robots that use language. Annual Review of Control, Robotics, and Autonomous Systems, 3(1):25–55, 2020.

[52] Cynthia Matuszek, Nicholas FitzGerald, Luke Zettlemoyer, Liefeng Bo, and Dieter Fox. A joint model of language and perception for grounded attribute learning, 2012.

[53] Felix Hill, Olivier Tieleman, Tamara von Glehn, Nathaniel Wong, Hamza Merzic, and Stephen Clark. Grounded language learning fast and slow, 2020.

[54] Thao Nguyen, Nakul Gopalan, Roma Patel, Matt Corsaro, Ellie Pavlick, and Stefanie Tellex. Robot object retrieval with contextual natural language queries, 2020.

[55] Jayant Krishnamurthy and Thomas Kollar. Jointly learning to parse and perceive: Connecting natural language to the physical world. Transactions of the Association for Computational Linguistics, 1:193–206, 12 2013.

[56] Patrick Jenkins, Rishabh Sachdeva, Gaoussou Youssouf Kebe, Padraig Higgins, Kasra Darvish, Edward Raff, Don Engel, John Winder, Francis Ferraro, and Cynthia Matuszek. Presentation and analysis of a multimodal dataset for grounded language learning, 2020.

[57] Jacob Arkin, Thomas Howard, Rohan Paul, and Nicholas Roy. Efficient grounding of abstract spatial concepts for natural language interaction with robot manipulators. The International Journal of Robotics Research, 37, 01 2016.

[58] Nisha Pillai, Karan K. Budhraja, and Cynthia Matuszek npillai. Improving grounded language acquisition efficiency using interactive labeling. 2016.

[59] Nisha Pillai and Cynthia Matuszek. Unsupervised selection of negative examples for grounded language learning. In AAAI, 2018.

[60] J. Brawer, O. Mangin, A. Roncone, S. Widder, and B. Scassellati. Situated human–robot collaboration: predicting intent from grounded natural language. In 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pages 827–833, 2018.

[61] C. Matuszek, D. Fox, and K. Koscher. Following directions using statistical machine translation. In 2010 5th ACM/IEEE International Conference on Human-Robot Interaction (HRI), pages 251–258, 2010.

[62] T. Kollar, Stefanie Tellex, D. Roy, and N. Roy. Grounding verbs of motion in natural language commands to robots. In ISER, 2010.

[63] Cynthia Matuszek, E. Herbst, Luke Zettlemoyer, and D. Fox. Learning to parse natural language commands to a robot control system. In ISER, 2012.

[64] Felix Duvallet. Natural language direction following for robots in unstructured unknown environments. 2015.

[65] Mohit Shridhar, Jesse Thomason, Daniel Gordon, Yonatan Bisk, Winson Han, Roozbeh Mottaghi, Luke Zettlemoyer, and Dieter Fox. Alfred: A benchmark for interpreting grounded instructions for everyday tasks. arXiv preprint arXiv:1912.01734, 2019.

[66] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015.

[67] Connor Shorten and T. Khoshgoftaar. A survey on image data augmentation for deep learning. Journal of Big Data, 6:1–48, 2019.

[68] Zhun Zhong, Liang Zheng, Guoliang Kang, Shaozi Li, and Yi Yang. Random erasing data augmentation, 2017.

[69] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d Alche-Buc, E. Fox, and R. Garnett, editors, Advances in Neural Information Processing Systems 32, pages 8024–8035. Curran Associates, Inc., 2019.

[70] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2017.

[71] Agrim Gupta, Piotr Dollár, and Ross Girshick. Lvis: A dataset for large vocabulary instance segmentation, 2019.

[72] Pramuditha Perera and Vishal M. Patel. Learning deep features for one-class classification. IEEE Transactions on Image Processing, 28(11):5450–5463, Nov 2019.