ABSTRACT

Detection of small objects in large swaths of imagery is one of the primary problems in satellite imagery analytics. While object detection in ground-based imagery has benefited from research into new deep learning approaches, transitioning such technology to overhead imagery is nontrivial. Among the challenges is the sheer number of pixels and geographic extent per image: a single DigitalGlobe satellite image encompasses > 64 km² and over 250 million pixels. Another challenge is that objects of interest are minuscule (often only ~ 10 pixels in extent), which complicates traditional computer vision techniques. To address these issues, we propose a pipeline (You Only Look Twice, or YOLT) that evaluates satellite images of arbitrary size at a rate of ≥ 0.5 km²/s. The proposed approach can rapidly detect objects of vastly different scales with relatively little training data over multiple sensors. We evaluate large test images at native resolution, and yield scores of F1 > 0.8 for vehicle localization. We further explore resolution and object size requirements by systematically testing the pipeline at decreasing resolution, and conclude that objects only ~ 5 pixels in size can still be localized with high confidence. Code is available at https://github.com/CosmiQ/yolt

KEYWORDS

Computer Vision, Satellite Imagery, Object Detection

1 INTRODUCTION

Computer vision techniques have made great strides in the past few years since the introduction of convolutional neural networks [5] in the ImageNet [13] competition. The availability of large, high-quality labelled datasets such as ImageNet [13], PASCAL VOC [2] and MS COCO [6] have helped spur a number of impressive advances in rapid object detection that run in near-real-time; three of the best are: Faster R-CNN [12], SSD [7], and YOLO [10] [11]. Faster R-CNN typically ingests 1000 × 600 pixel images, whereas SSD uses 300 × 300 or 512 × 512 pixel input images, and YOLO runs on either 416 × 416 or 544 × 544 pixel inputs. While the performance of all these frameworks is impressive, none can come remotely close to ingesting the ~ 16,000 × 16,000 input sizes typical of satellite imagery. Of these three frameworks, YOLO has demonstrated the greatest inference speed and highest score on the PASCAL VOC dataset. The authors also showed that this framework is highly transferrable to new domains by demonstrating superior performance to other frameworks (i.e., SSD and Faster R-CNN) on the Picasso Dataset [3] and the People-Art Dataset [1]. Due to the speed, accuracy, and flexibility of YOLO, we accordingly leverage this system as the inspiration for our satellite imagery object detection framework.

The application of deep learning methods to traditional object detection pipelines is non-trivial for a variety of reasons. The unique aspects of satellite imagery necessitate algorithmic contributions to address challenges related to the spatial extent of foreground target objects, complete rotation invariance, and a large scale search space. Excluding implementation details, algorithms must adjust for:

Small spatial extent In satellite imagery objects of interest are often very small and densely clustered, rather than the large and prominent subjects typical in ImageNet data. In the satellite domain, resolution is typically defined as the ground sample distance (GSD), which describes the physical size of one image pixel. Commercially available imagery varies from 30 cm GSD for the sharpest DigitalGlobe imagery, to 3–4 meter GSD for Planet imagery. This means that for small objects such as cars each object will be only ~ 15 pixels in extent even at the highest resolution.

Complete rotation invariance Objects viewed from overhead can have any orientation (e.g. ships can have any heading between 0 and 360 degrees, whereas trees in ImageNet data are reliably vertical).

Training example frequency There is a relative dearth of training data (though efforts such as SpaceNet¹ are attempting to ameliorate this issue)

Ultra high resolution Input images are enormous (often hundreds of megapixels), so simply downsampling to the input size required by most algorithms (a few hundred pixels) is not an option (see Figure 1).

The contribution in this work specifically addresses each of these issues separately, while leveraging the relatively constant distance from sensor to object, which is well known and is typically ~ 400 km. This coupled with the nadir facing sensor results in consistent pixel size of objects.

Section 2 details in further depth the challenges faced by standard algorithms when applied to satellite imagery. The remainder of this work is broken up to describe the proposed contributions as follows. To address small, dense clusters, Section 3.1 describes a new, finer-grained network architecture. Sections 3.2 and 3.3 detail our method for splitting, evaluating, and recombining large test images of arbitrary size at native resolution. With regard to rotation invariance and small labelled training dataset sizes, Section 4 describes data augmentation and size requirements. Finally, the performance of the algorithm is discussed in detail in Section 6.

¹https://aws.amazon.com/public-datasets/spacenet/
2 RELATED WORK

Deep learning approaches have proven effective for ground-based object detection, though current techniques are often still suboptimal for overhead imagery applications. For example, small objects in groups, such as flocks of birds present a challenge [10], caused in part by the multiple downsampling layers of all three convolutional network approaches listed above (YOLO, SDD, Faster-RCNN). Further, these multiple downsampling layers result in relatively coarse features for object differentiation; this poses a problem if objects of interest are only a few pixels in extent. For example, consider the default YOLO network architecture, which downsamples by a factor of 32 and returns a $13 \times 13$ prediction grid; this means that object differentiation is problematic if object centroids are separated by less than 32 pixels. Accordingly we implement a unique network architecture with a denser final prediction grid. This improves performance by yielding finer grained features to help differentiate between classes. This finer prediction grid also permits classification of smaller objects and denser clusters.

Another reason object detection algorithms struggle with satellite imagery is that they have difficulty generalizing objects in new or unusual aspect ratios or configurations [10]. Since objects can have arbitrary heading, this limited range of invariance to rotation is troublesome. Our approach remedies this complication with rotations and augmentation of data. Specifically, we rotate training images about the unit circle to ensure that the classifier is agnostic to object heading, and also randomly scale the images in HSV (hue-saturation-value) to increase the robustness of the classifier to varying sensors, atmospheric conditions, and lighting conditions.

In advanced object detection techniques the network sees the entire image at train and test time. While this greatly improves background differentiation since the network encodes contextual (background) information for each object, the memory footprint on typical hardware (NVIDIA Titan X GPUs with 12GB RAM) is infeasible for a 256 megapixel image.

2 RELATED WORK

Deep learning approaches have proven effective for ground-based object detection, though current techniques are often still suboptimal for overhead imagery applications. For example, small objects in groups, such as flocks of birds present a challenge [10], caused in part by the multiple downsampling layers of all three convolutional network approaches listed above (YOLO, SDD, Faster-RCNN). Further, these multiple downsampling layers result in relatively coarse features for object differentiation; this poses a problem if objects of interest are only a few pixels in extent. For example, consider the default YOLO network architecture, which downsamples by a factor of 32 and returns a $13 \times 13$ prediction grid; this means that object differentiation is problematic if object centroids are separated by less than 32 pixels. Accordingly we implement a unique network architecture with a denser final prediction grid. This improves performance by yielding finer grained features to help differentiate between classes. This finer prediction grid also permits classification of smaller objects and denser clusters.

Another reason object detection algorithms struggle with satellite imagery is that they have difficulty generalizing objects in new or unusual aspect ratios or configurations [10]. Since objects can have arbitrary heading, this limited range of invariance to rotation is troublesome. Our approach remedies this complication with rotations and augmentation of data. Specifically, we rotate training images about the unit circle to ensure that the classifier is agnostic to object heading, and also randomly scale the images in HSV (hue-saturation-value) to increase the robustness of the classifier to varying sensors, atmospheric conditions, and lighting conditions.

In advanced object detection techniques the network sees the entire image at train and test time. While this greatly improves background differentiation since the network encodes contextual (background) information for each object, the memory footprint on typical hardware (NVIDIA Titan X GPUs with 12GB RAM) is infeasible for a 256 megapixel image.

Figure 1: DigitalGlobe $8 \times 8$ km ($\sim 16,000 \times 16,000$ pixels) image at 50 cm GSD near the Panama Canal. One $416 \times 416$ pixel sliding window cutout is shown in red. For an image this size, there are $\sim 1500$ unique cutouts.

Figure 2: Challenges of the standard object detection network architecture when applied to overhead vehicle detection. Each image uses the same standard YOLO architecture model trained on $416 \times 416$ pixel cutouts of cars from the COWC dataset. Left: Model applied to a large $4000 \times 4000$ pixel test image downsampled to a size of $416 \times 416$; none of the 1142 cars in this image are detected. Right: Model applied to a small $416 \times 416$ pixel cutout; the excessive false negative rate is due to the high density of cars that cannot be differentiated by the $13 \times 13$ grid.

3 YOU ONLY LOOK TWICE

In order to address the limitations discussed in Section 2, we implement an object detection framework optimized for overhead imagery: You Only Look Twice (YOLT). We extend the Darknet neural network framework [9] and update a number of the C libraries to enable analysis of geospatial imagery and integrate with external python libraries. We opt to leverage the flexibility and large
3.1 Network Architecture

To reduce model coarseness and accurately detect dense objects (such as cars or buildings), we implement a network architecture that uses 22 layers and downsamples by a factor of 16. Thus, a 416 x 416 pixel input image yields a 26 x 26 prediction grid. Our architecture is inspired by the 30-layer YOLO network, though this new architecture is optimized for small, densely packed objects. The dense grid is unnecessary for diffuse objects such as airports, but crucial for high density scenes such as parking lots (see Figure 2). To improve the fidelity of small objects, we also include a passthrough layer (described in [11], and similar to identity mappings in ResNet [4]) that concatenates the final 52 x 52 layer onto the last convolutional layer, allowing the detector access to finer grained features of this expanded feature map.

Each convolutional layer save the last is batch normalized with a leaky rectified linear activation, save the final layer that utilizes a linear activation. The final layer provides predictions of bounding boxes and classes, and has size: \( N_f = N_{boxes} \times (N_{classes} + 5) \), where \( N_{boxes} \) is the number of boxes per grid (5 by default), and \( N_{classes} \) is the number of object classes [10].

3.2 Test Procedure

At test time, we partition testing images of arbitrary size into manageable cutouts and run each cutout through our trained model. Partitioning takes place via a sliding window with user defined bin sizes and overlap (15% by default), see Figure 4. We record the position of each sliding window cutout by naming each cutout according to the schema:

```
ImageName|row_column_height_width.ext
```

For example:

```
panama50cm|11370_1180_416_416.tif
```

3.3 Post-Processing

Much of the utility of satellite (or aerial) imagery lies in its inherent ability to map large areas of the globe. Thus, small image chips...
Training data is collected from small chips of large images from very effective.

Section 6.2 that using two different detectors at different scales is of very different scales (e.g., airplanes, boats, building footprints, cars, and airports. For objects identified for each object. We initially focus on five categories: platforms. Labels are comprised of a bounding box and category for each object. Initially we attempt to train a single classifier to recognize all five categories listed above, both vehicles and infrastructure. We note a number of spurious airport detections in this example (see Figure 6),

4 TRAINING DATA

Training data is collected from small chips of large images from three sources: DigitalGlobe satellites, Planet satellites, and aerial platforms. Labels are comprised of a bounding box and category identifier for each object. We initially focus on five categories: airplanes, boats, building footprints, cars, and airports. For objects of very different scales (e.g. airplanes vs airports) we show in Section 6.2 that using two different detectors at different scales is very effective.

Cars The Cars Overhead with Context (COWC) [8] dataset is a large, high quality set of annotated cars from overhead imagery collected over multiple locales. Data is collected via aerial platforms, but at a nadir view angle such that it resembles satellite imagery. The imagery has a resolution of 15 cm GSD that is approximately double the current best resolution of commercial satellite imagery (30 cm GSD for DigitalGlobe). Accordingly, we convolve the raw imagery with a Gaussian kernel and reduce the image dimensions by half to create the equivalent of 30 cm GSD images. Labels consist of simply a dot at the centroid of each car, and we draw a 3 meter bounding box around each car for training purposes. We reserve the largest geographic region (Utah) for testing, leaving 13,303 labelled training cars.

Building Footprints The second round of SpaceNet data consists of 30 cm GSD DigitalGlobe imagery and labelled building footprints over four cities: Las Vegas, Paris, Shanghai, and Khartoum. The labels are precise building footprints, which we transform into bounding boxes encompassing 90% of the extent of the footprint. Image segmentation approaches show great promise for this challenge; nevertheless, we explore YOLT performance on building outline detection, acknowledging that since YOLT outputs bounding boxes it will never achieve perfect building footprint detection for complex building shapes. Between the four cities there are 221,336 labelled buildings.

Airplanes We label eight DigitalGlobe images over airports for a total of 230 objects in the training set.

Boats We label three DigitalGlobe images taken over coastal regions for a total of 556 boats.

Airports We label airports in 37 Planet images for training purposes, each with a single airport per chip. For objects the size of airports, some downsampling is required, as runways can exceed 1000 pixels in length even in low resolution Planet imagery; we therefore downsample Planet imagery by a factor of four for training purposes.

The raw training datasets for airplanes, airports, and watercraft are quite small by computer vision standards, and a larger dataset may improve the inference performance detailed in Section 6.

We train with stochastic gradient descent and maintain many of the hyper parameters of [11]: 5 boxes per grid, an initial learning rate of $10^{-3}$, a weight decay of 0.0005, and a momentum of 0.9. Training takes 2 – 3 days on a single NVIDIA Titan X GPU.

5 TEST IMAGES

To ensure evaluation robustness, all test images are taken from different geographic regions than training examples. For cars, we reserve the largest geographic region of Utah for testing, yielding 19,807 test cars. Building footprints are split 75/25 train/test, leaving 73,778 test footprints. We label four airport test images for a total of 74 airplanes. Four boat images are labelled, yielding 771 test boats. Our dataset for airports is smaller, with ten Planet images used for training purposes. We reserve the largest geographic region (Utah) for testing, leaving 13,303 labelled training cars.

6 OBJECT DETECTION RESULTS

6.1 Universal Classifier Object Detection Results

Initially, we attempt to train a single classifier to recognize all five categories listed above, both vehicles and infrastructure. We note a number of spurious airport detections in this example (see Figure 6),
Table 2: Train/Test Split

| Object Class | Training Examples | Test Examples |
|--------------|-------------------|---------------|
| Airport*     | 37                | 10            |
| Airplane*    | 230               | 74            |
| Boat*        | 556               | 100           |
| Car†         | 19,807            | 13,303        |
| Building†    | 221,336           | 73,778        |

* Internally Labelled
† External Dataset

Figure 6: Poor results of the universal model applied to DigitalGlobe imagery at two different scales (200m, 1500m). Airplanes are in red. The cyan boxes mark spurious detections of runways, caused in part by confusion from small scale linear structures such as highways.

6.2 Scale Confusion Mitigation

There are multiple ways one could address the false positive issues noted in Figure 6. Recall from Section 4 that for this exploratory work our training set consists of only a few dozen airports, far smaller than usual for deep learning models. Increasing this training set size might improve our model, particularly if the background is highly varied. Another option would be to use post-processing to remove any detections at the incorrect scale (e.g. an airport with a size of ~50 meters). Another option is to simply build dual classifiers, one for each relevant scale.

We opt to utilize the scale information present in satellite imagery and run two different classifiers: one trained for vehicles + buildings, and the other trained only to look for airports. Running the second airport classifier on down sampled images has a minimal impact on runtime performance, since in a given image there are approximately 100 times more 200 meter chips than 2000 meter chips.

Figure 7: Car detection performance on a 600 × 600 meter aerial image over Salt Lake City (ImageID = 21) at 30 cm GSD with 1389 cars present. False positives are shown in red, false negatives are yellow, true positives are green, and blue rectangles denote ground truth for all true positive detections. F1 = 0.95 for this test image, and GPU processing time is < 1 second.

6.3 Dual Classifier Results

For large validation images, we run the classifier at two different scales: 200m, and 2500m. The first scale is designed for vehicles and buildings, and the larger scale is optimized for large infrastructure such as airports. We break the validation image into appropriately sized image chips and run each image chip on the appropriate classifier. The myriad results from the many image chips and multiple classifiers are combined into one final image, and overlapping detections are merged via non-maximal suppression. We find a detection probability threshold of between 0.3 and 0.4 yields the highest F1 score for our validation images.

We define a true positive as having an intersection over union (IOU) of greater than a given threshold. An IOU of 0.5 is often used as the threshold for a correct detection, though as in Equation 5 of ImageNet [13] we select a lower threshold for vehicles since we are dealing with very small objects. For SpaceNet building footprints and airports we use an IOU of 0.5.

Table 3 displays object detection performance and speed over all test images for each object category. YOLT performs relatively well on airports, airplanes, and boats, despite small training set sizes. YOLT is not optimized for building footprint extraction, though performs somewhat competitively on the SpaceNet dataset; the top score on the recent SpaceNet challenge achieved an F1 score of 0.69².

²https://spacenetchallenge.github.io/Competitions/Competition2.html
Figure 8: YOLT classifier applied to a SpaceNet DigitalGlobe 50 cm GSD image containing airplanes (blue), boats (red), and runways (orange). In this image we note the following F1 scores: airplanes = 0.83, boats = 0.84, airports = 1.0.

Table 3: YOLT Performance and Speed

| Object Class | F1 Score | Run Time (km²/min) |
|--------------|----------|--------------------|
| Car†         | 0.90 ± 0.09 | 32                 |
| Airplane *   | 0.87 ± 0.08 | 32                 |
| Boat         | 0.82 ± 0.07 | 32                 |
| Building *   | 0.61 ± 0.15 | 32                 |
| Airport *    | 0.91 ± 0.14 | 6000               |

† IOU = 0.25
* IOU = 0.5

while the YOLT score of 0.61 puts it in the top 3. We report inference speed in terms of GPU time to run the inference step. Inference runs rapidly on the GPU, at ~ 50 frames per second. Currently, pre-processing (i.e., splitting test images into smaller cutouts) and post-processing (i.e., stitching results back into one global image) is not fully optimized and is performed on the CPU, which adds a factor of ~ 2 to run time. The inference speed translates to a runtime of < 6 minutes to localize all vehicles in an area of the size of Washington DC, and < 2 seconds to localize airports over this area. DigitalGlobe’s WorldView3 satellite³ covers a maximum of 680,000 km² per day, so at YOLT inference speed a 16 GPU cluster would provide real-time inference on satellite imagery.

6.4 Detailed Performance Analysis

The large test set of ~ 20,000 cars in the nine Utah images of the COWC dataset enables detailed performance analyses. The majority of the cars (> 13,000) lie in the image over central Salt Lake City so we split this image into sixteen smaller 600 × 600 meter regions to even out the number of cars per image. We remove one test scene that has only 61 cars in the scene, leaving 23 test scenes, with mean count per test image of 1130 ± 540. We apply a YOLT model trained to find cars on these test scenes.

In Figure 9 we display the F1 score for each scene, along with the car count accuracy. Total car count in a specified region may be a more valuable metric in the commercial realm than F1 score. Accordingly, we compute the number of predicted cars for each scene as a fraction of ground truth number (F_c = N_{predicted}/N_{truth}). Like the F1 score, a value of 1.0 denotes perfect prediction for the fractional car count metric. The COWC [8] authors sought to count (rather than localize) the number of cars in test images, and achieved an error of 0.19%. Total count error for YOLT on the COWC data is 0.90%.

Inspection of Figure 9 reveals that the F1 score and ground truth fraction are quite high for typical urban scenes, (e.g. ImageID = 21 shown in Figure 7). The worst outlier in Figure 9 is ImageID = 2, with an F1 score of 0.67, and 2860 cars present. This location corresponds to an automobile junkyard, an understandably difficult region to analyze.

³http://worldview3.digitalglobe.com
7 RESOLUTION PERFORMANCE STUDY

The uniformity of object sizes in the COWC [8] dataset enables a detailed resolution study. To study the effects of resolution on object detection, we convolve the raw 15 cm imagery with a Gaussian kernel and reduce the image dimensions to create additional training and testing corpora at [0.30, 0.45, 0.60, 0.75, 0.90, 1.05, 1.20, 1.50, 1.80, 2.10, 2.40, 3.00] meters.

Initially, we test the multi-resolution test data on a single model (trained at 0.30 meters), and in Figure 11 demonstrate that the ability of this model to extrapolate to multiple resolutions is poor. Subsequently, we train a separate model for each resolution, for thirteen models total. Creating a high quality labelled dataset at low resolution (2.4m GSD, for example) is only possible because we downsample from already labelled high resolution 15 cm data; typically low resolution data is very difficult to label with high accuracy.

For objects ~ 3 meters in size we observe from Figure 13 that object detection performance degrades from $F_1 = 0.92$ for objects 20 pixels in size to $F_1 = 0.27$ for objects 1 pixel in size, with a mean error of 0.09. Interestingly, the $F_1$ score only degrades by only $\approx 5\%$ as objects shrink from 20 to 5 pixels in size (0.15m to 0.60m GSD). At least for cars viewed from overhead, one can conclude that object sizes of $\geq 5$ pixels yield object detection scores of $F_1 > 0.85$. The curves of Figure 11 degrade far faster than Figures 13 and 14, illustrating that a single model fit at high resolution is inferior to a series of models trained at each respective resolution.

8 CONCLUSIONS

Object detection algorithms have made great progress as of late in localizing objects in ImageNet style datasets. Such algorithms are rarely well suited to the object sizes or orientations present in satellite imagery, however, nor are they designed to handle images with hundreds of megapixels.

To address these limitations we implemented a fully convolutional neural network pipeline (YOLT) to rapidly localize vehicles, buildings, and airports in satellite imagery. We noted poor results from a combined classifier due to confusion between small and large features, such as highways and runways. Training dual classifiers at different scales (one for buildings/vehicles, and one for infrastructure), yielded far better results.

This pipeline yields an object detection $F_1$ score of $\approx 0.6 - 0.9$, depending on category. While the $F_1$ scores may not be at the level many readers are accustomed to from ImageNet competitions, object detection in satellite imagery is still a relatively nascent field and has unique challenges. In addition, our training dataset for most categories is relatively small for supervised learning methods, and the $F_1$ scores could possibly be improved with further post-processing of detections.

We also demonstrated the ability to train on one sensor (e.g. DigitalGlobe), and apply our model to a different sensor (e.g. Planet). We show that at least for cars viewed from overhead, object sizes
of ≥ 5 pixels yield object detection scores of $F_1 > 0.85$. The detection pipeline is able to evaluate satellite and aerial images of arbitrary input size at native resolution, and processes vehicles and buildings at a rate of ≈ 30 km$^2$ per minute, and airports at a rate of ≈ 6,000 km$^2$ per minute. At this inference speed, a 16 GPU cluster could provide real-time inference on the DigitalGlobe WorldView3 satellite feed.

ACKNOWLEDGMENTS

We thank Karl Ni for very helpful comments.

REFERENCES

[1] Hongping Cai, Qi Wu, Tadeo Corradi, and Peter Hall. 2015. The Cross-Depiction Problem: Computer Vision Algorithms for Recognising Objects in Artwork and in Photographs. CoRR abs/1505.00110 (2015). http://arxiv.org/abs/1505.00110

[2] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. 2010. The Pascal Visual Object Classes (VOC) Challenge. International Journal of Computer Vision 88, 2 (June 2010), 303–338.

[3] Shary Ginosa, Daniel Has, Timothy Brown, and Jitendra Malik. 2014. Detecting People in Cubist Art. CoRR abs/1409.6235 (2014). http://arxiv.org/abs/1409.6235

[4] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2015. Deep Residual Learning for Image Recognition. CoRR abs/1512.03385 (2015). http://arxiv.org/abs/1512.03385

[5] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. ImageNet Classification with Deep Convolutional Neural Networks. In Advances in Neural Information Processing Systems 25. F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger (Eds.). Curran Associates, Inc., 1097–1105. http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf

[6] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, Lubomir D. Bourdev, Ross B. Girshick, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. 2014. Microsoft COCO: Common Objects in Context. CoRR abs/1405.0312 (2014). http://arxiv.org/abs/1405.0312

[7] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott E. Reed, Cheng-Yang Fu, and Alexander C. Berg. 2015. SSD: Single Shot MultiBox Detector. CoRR abs/1512.02325 (2015). http://arxiv.org/abs/1512.02325

[8] T. Nathan Mundhenk, Goran Konjevod, Wesan A. Salda, and Koh Boakye. 2016. A Large Contextual Dataset for Classification, Detection and Counting of Cars with Deep Learning. CoRR abs/1609.04453 (2016). http://arxiv.org/abs/1609.04453

[9] J. Redmon. 2013-2017. Darknet: Open source neural networks in c. http://pjreddie.com/darknet/ (2015). http://arxiv.org/abs/1506.02640

[10] Joseph Redmon, Santosh Kumar Divvala, Ross B. Girshick, and Ali Farhadi. 2015. You Only Look Once: Unified, Real-Time Object Detection. CoRR abs/1506.02640 (2015). http://arxiv.org/abs/1506.02640

[11] Joseph Redmon and Ali Farhadi. 2016. YOLO9000: Better, Faster, Stronger. CoRR abs/1612.08242 (2016). http://arxiv.org/abs/1612.08242

[12] Shaoqing Ren, Kaiming He, Ross B. Girshick, and Jian Sun. 2015. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. CoRR abs/1506.01497 (2015). http://arxiv.org/abs/1506.01497

[13] 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. 2015. ImageNet Large Scale Visual Recognition Challenge. International Journal of Computer Vision (IJCV) 115, 3 (2015), 211–252.

https://doi.org/10.1007/s11263-015-0816-y

Figure 13: Object detection F1 score for ground sample distances of 0.15 – 3.0 meters (bottom axis), corresponding to car size of 20 – 1 pixel(s) (top axis). At each of the thirteen resolutions we evaluate test scenes with a unique model trained at that resolution. The 23 thin lines display the performance of the individual test scenes; most of these lines are tightly clustered about the mean, denoted by the blue dashed line. The red band displays ±1 STD. We fit a piecewise linear model to the data, shown as the dotted cyan line. Below the inflection point (large cyan dot) of 0.61 meters (corresponding to a car size of 5 pixels) the F1 score degrades slowly with a slope of $\Delta F_1/\Delta GSD = -0.10$; between 0.60 m and 3.0 m GSD the slope is steeper at $-0.26$. The F1 scores at 0.15 m, 0.60 m, and 3.0 m GSD are 0.92, 0.87, and 0.27, respectively.

Figure 14: Fraction of predicted number of cars to ground truth, with a unique model for each resolution (bottom axis) and object pixel size (top axis). A fraction of 1.0 means that the correct number of cars was predicted, while if the fraction is below 1.0 too few cars were predicted. The thin bands denote the performance of the 23 individual scenes, with the dashed blue line showing the weighted mean and the red band displaying ±1 STD. We fit a piecewise linear model to the data, shown as the dotted cyan line. Below the inflection point (large cyan dot) of 0.86 meters the slope is essentially flat with a slope of $-0.03$; between 0.87 m and 3 m GSD the slope is steeper at $-0.20$. For resolutions sharper than 0.86 meters the predicted number of cars is within 4% of ground truth.