Collecting Image Datasets with a Quadcopter

Saif Alabachi, Gita Sukthankar

Department of Electrical and Computer Engineering
University of Central Florida; Orlando, Florida

s.mohammed@knights.ucf.edu, gitars@eecs.ucf.edu

ABSTRACT

Although generic object detection models exist, it is important to train mobile robots to detect and recognize specific objects that are unique to an environment. Transfer learning can be used to reduce the number of new images required to learn the new object instances by leveraging more general existing models. This paper describes our user interface for collecting custom image datasets with a quadcopter and performing object instance labeling and annotation semi-autonomously. Collecting images with marked objects is a time-consuming process and usually done under heavy human supervision. We evaluated our dataset by detecting the annotated objects in the real-time quadcopter video feed.

Keywords

Image Dataset Construction, Transfer Learning, Human Robot Interaction, Semi-autonomous agent, Quadcopter.

1. INTRODUCTION

Convolutional neural networks have achieved outstanding performance on object detection methods but depend on the existence of large datasets with millions of annotated images [1]. However, image dataset construction remains a time-consuming task that requires substantial effort even with the availability of helpful tools. Our work addresses the problem of collecting and annotating a dataset of object images using a quadcopter. We demonstrate that we can use the collected images to create customized object detectors for important landmarks in our indoor environment. The quadcopter serves a dual purpose: not only is it used during the image collection process but also it can utilize the customized object detectors.

Customized object detectors can be trained with fewer images using transfer learning. Transferring image representations reduces the time and data needed for training, because the models leverage features learned from the original dataset. It is possible to fine tune or transfer a pre-trained model and still obtain satisfying detection results without the need to relearn the entire network. For instance, Oquab et al. addressed this problem in object and action classification using a transfer learning model trained on different datasets [2].

However, dataset construction and annotation require major manual effort even with the availability of helpful tools. Our aim is to automate this process and reduce the need for human monitoring. For this project we developed a sketch based user interface system to guide the quadcopter. The user simply sketches a bounding box around the object of interest, and the quadcopter collects images semi-autonomously with minimal guidance from the human. To add variance to the dataset while reducing the flight time, we added image filtering capabilities to the user interface to augment the dataset with additional synthetic images with changes to brightness, contrast, zooming, and rotation. For transfer learning, we selected a state of the art object detection model: the SSD [3] MobileNet [4] architecture for compute constrained devices (built on VGG-16 [5]).

2. RELATED WORK

The availability of large image datasets has yielded dramatic improvements in image classification. Important datasets include: CIFAR-10 with 6000 examples of 10 classes, CIFAR100 with 600 examples of 100 non-overlapping classes [6], the Lotus Hill [7] dataset with 50,000 images, and ImageNet [8] with almost 15 million high-resolution images in 22,000 categories. In ImageNet, the images are collected from the web and labeled using Amazon's Mechanical Turk [1]. However these datasets typically lack bounding boxes localizing the object instances, and the ones that contain bounding boxes are usually human-annotated. Examples include COCO [9] which has about 80 object categories with over 1.5 million object instances, and PascalVOC [10] with 20 classes and 27,450 annotated objects.

Some tools have been created to help researchers label images into segments. LabelMe [11] is a web-based tool that allows easy image annotation and instant sharing. One collection strategy is to gather natural samples from the Internet using query patterns to generate the desired image dataset [12]. It is also possible to analyze the image and its text-annotation in order to select a ground-figure segmentation and to use this information to classify segments into visual categories [13].

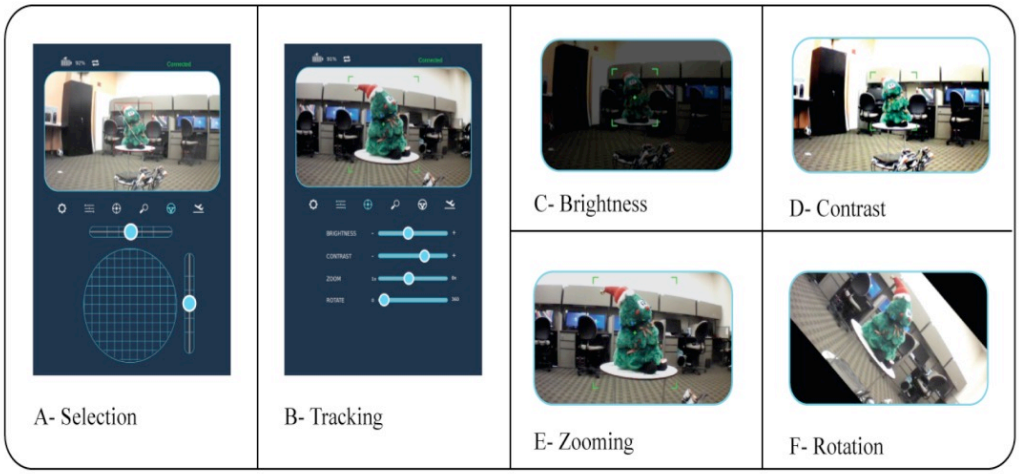

Fig. 1. A. The user selects the object after quadcopter take off. B. The tracking process starts and the quadcopter flies autonomously to free the user's attention. C. The brightness filter is applied before saving the candidate image. D. Contrast filter application. E. Zooming filter application. F. Rotation filter application.

The COCO challenge [9] was created to advance the state of the art in object recognition by making it advantageous to combine object recognition with scene understanding. For precise object localization, COCO objects are labeled using per-instance segmentations; the dataset contains photos of 91 objects with crowdsourced annotations labeled using a novel user interface for category detection [2]. Our research improves on existing tools by using a quadcopter for image collection and adding functions to the flight interface to synthetically augment the dataset.

3. METHOD

In this section, we describe the procedure used to collect our customizable annotated-object image dataset. Our experiments were performed on the commercially available Parrot Augmented Reality (AR) Drone Version 2. This drone has two cameras: one front-mounted HD camera and a downward facing QVGA camera. For our experiments, we extended an early version of our user interface described in [14]; a video demo of the original system can be viewed at: https://youtu.be/ErA2111xjzMl.

3.1 Dataset Construction

First the human manually flies the quadcopter to the object of interest and then sketches a circle around the object to initialize the four coordinates of the bounding box (x min, y min x max , y max). The selection should only include a single object to match the original training COCO dataset (see Figure 1 A).

The bounding box is sent to our system to initialize autonomous navigation. An object tracker is used to calculate the bounding box in subsequent frames which are saved to the image dataset.

Sometimes, due to the network delay, undesired annotations may occur that need to be eliminated from the dataset before the learning process. The system collects one candidate image each second to allow enough time for the filtering operation to be applied before storing the resultant frame. We believe that our platform can also be used to create action image datasets following the same procedure.

3.2 Tracking

For tracking, we evaluated several online trackers available within openCV and one hybrid tracker (combining offline and online tracking). Some of the trackers failed since they were not designed for a moving camera, and others had problems achieving real-time performance on a mobile computer. However, the adaptive correlation filter, MOSSE [15], was found to be stable and capable of handling the 30 frames per second generated by the Parrot ARDrone 2.0 camera.

3.3 Image Filters

Before storing the candidate sample, the annotated frames can be processed using image filtering operations in order to create more variation in the dataset. These filters are accessed directly through the user interface and include brightness, contrast, rotation, and zooming. Rotation is useful as it is not possible to hover with a tilt orientation while capturing high quality images, and artificially zooming preserves the battery life by reducing the quadcopter movement. Figure 1 B, C, D, E, and F show examples of images captured with these filters.

3.4 Semi-autonomous Navigation

After the initial bounding box is drawn, the quadcopter starts flying autonomously, and the system enters a visual dataset collection mode, acquiring data at a rate of 1 fps. The quadcopter

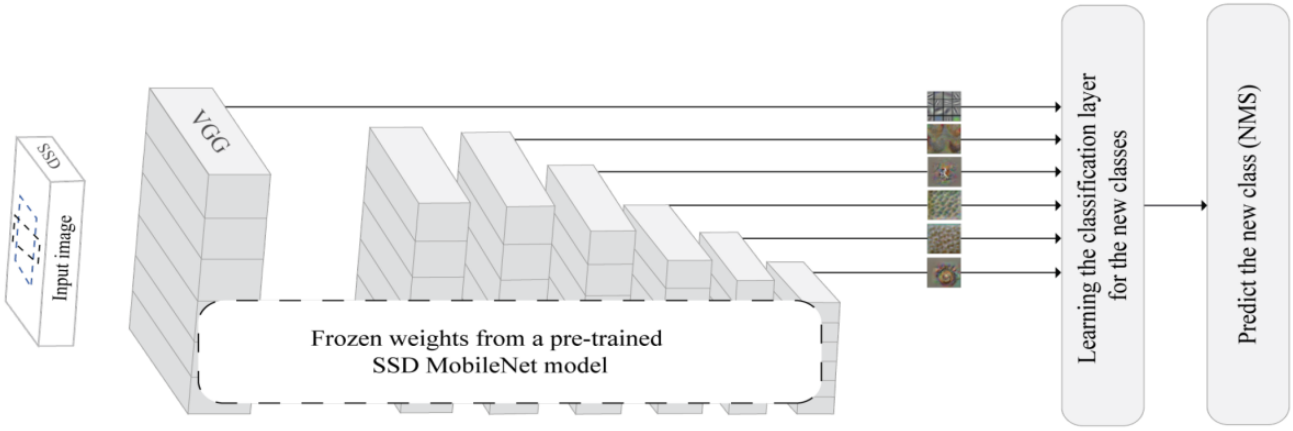

Fig. 2. Transfer learning from the SSD MobileNet architecture

modifies its yaw angle and altitude to track the object designated by the user. The x-axis error between the object centroid and canvas center is used to estimate the orientation angle, and the y-axis error is used to estimate the quadcopter's altitude.

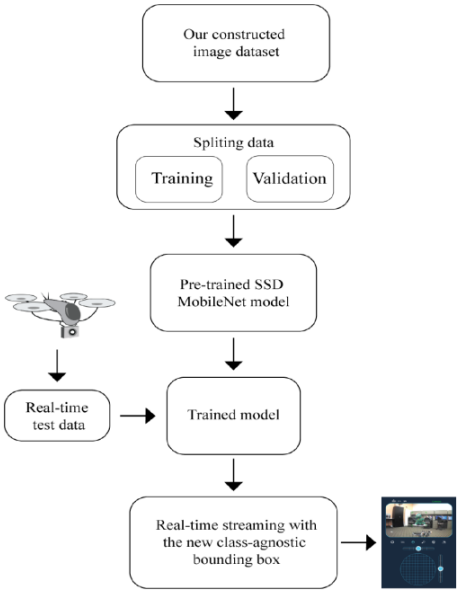

Fig. 3. Flowchart illustrates the transfer learning phase on the new constructed dataset and the testing phase using the video streaming from the quadcopter.

The errors are transmitted to a PD (proportional-derivative) controller with gains K_p and K_d set to 0.25. The quadcopter uses its inertial sensors to monitor roll Φ, pitch Θ, yaw ψ, rotational speed Ψ and the vertical velocity ζ; controls are issued using a series of ROS Twist commands u = (Φ, Θ, ζ, Ψ) □ $[-1, 1]^4$ at a frequency of 100Hz. Our interface is capable of eliminating undesired photos by comparing the correlation percentage to a predefined threshold; as long as this percentage exceeds the specified threshold, the agent continues photographing the tracked object, else it stops. This scenario was inspired by ImageNet where the dataset contains images with a single centered object.

3.5 Transfer Learning

Transfer learning is beneficial because our dataset is small compared to the generic datasets. To reduce the time required for training, all the model weights are frozen except the last classification layer. Figure 2 shows the transfer learning process. During the training phase, we transfer weights trained on the SSD architecture which is a feed-forward convolutional network; it is implemented using the MobileNet model which is based on depth-wise separable convolutions. These weights were trained on the COCO dataset before being frozen; the collected dataset is then used to train the classifier layer for the new classes. Since SSD uses a different aspect ratio for different scales (8x8 and 4x4), our system collects several frames from different locations using both zooming and navigation.

Table 1. Object detection performance on new objects

Constructed Dataset Prediction Results	
Class	**Percentage**
coffee machine	84%
Christmas toy	53%
potted plant	81%
tissue box	98%

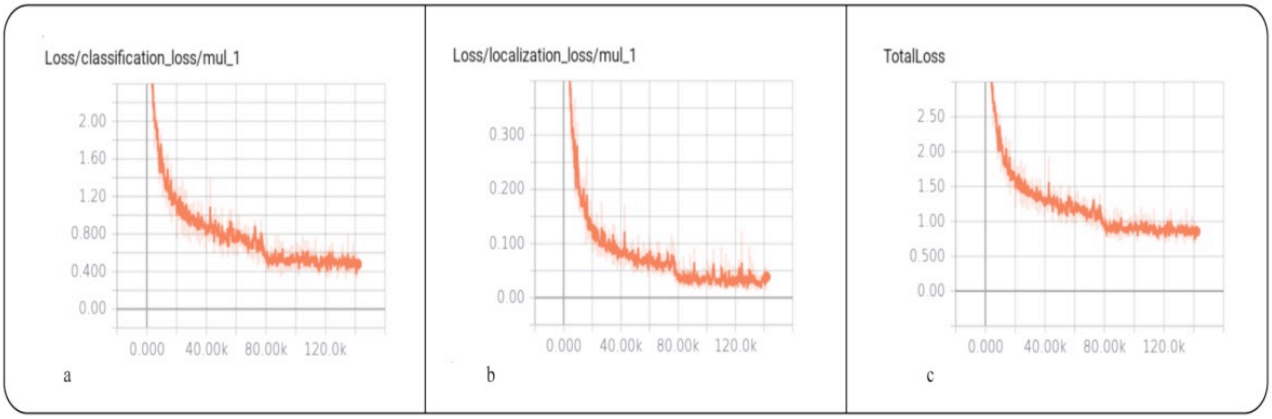

Fig. 4. a. Classification Loss. b. Localization Loss. c. Total Loss = (Confidence Loss + $\alpha*$ Localization Loss)

4. RESULTS

For our experiments, we collected a dataset with images for four classes of which three are new and one already existed in the COCO dataset. The classes are: 1) coffee machine 2) Xmas toy 3) potted plant and 4) tissue box. Our image dataset contains 270 bounding box annotated images for each class with one class per

image. The examples always have the object located near the center of the frame. Our test samples consist of 640×360 frames gathered from the real-time video streaming of the quadcopter frontal camera. The transfer learning process required two days to reach an average total loss (Confidence Loss + α Location Loss) < 0.9. α is a parameter reducing the location loss function by bringing the predictions closer to the ground truth. Our training configuration is batch size = 60, learning rate = 0.004 with 0.95 decay factor. Figure 4 shows the loss functions in the graphs a, b, and c.

The architecture is SSD MobileNet, and the object detection API provided by the Tensorflow community is used to detect the new classes. We used weights extracted from a network trained on the COCO dataset before the classification layer. The new classifier is trained on the constructed dataset. Figure 5 shows the object detection model after applying transfer learning on the constructed dataset. As shown in the left figure, the potted plant is an existing class, but the model can't detect it with confidence > 50%, whereas the trained model is able to detect all the new classes after one day of transfer learning.

In the testing phase, we evaluate our dataset annotation accuracy by obtained the classifier performance on predicting the class of the objects appearing in the test data. The classifier predictions are shown in Table I. Figure 3 illustrates our experiment. The bounding box surrounding the objects in the live stream indicates successful real-time object detection.

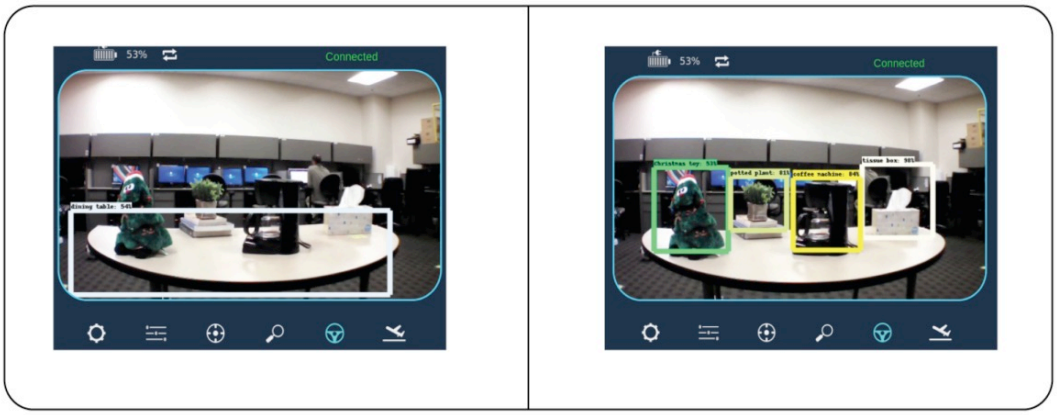

Fig. 5. To the left is the object detection model trained on the COCO dataset. Christmas toy, tissue box, and coffee machine are new classes whereas potted plant is an existent class. The picture to the right is the detection result after the completion of transfer learning.

CONCLUSION

This paper presents a novel approach to constructing an annotated object image dataset using a semi-autonomous quadcopter that gathers multiple viewpoints of a target object and applies image filters to create a synthetically augmented dataset for training customized object detectors using transfer learning on a CNN model. Our platform is capable of capturing high quality images of a fixed or moving object using a friendly user interface that can be launched from a mobile device. We demonstrate that the customized object detectors trained with the semi-autonomously constructed dataset perform well at detecting objects viewed through the quadcopter video feed in real time.

5. REFERENCES

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classifica-tion with deep convolutional neural networks," in Advances in Neural Information Processing Systems, 2012, pp. 1097–1105.

[2] M. Oquab, L. Bottou, I. Laptev, and J. Sivic, "Learning and transferring mid-level image representations using convolutional neural networks," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2014, pp. 1717–1724.

[3] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "SSD: Single shot multibox detector," in European Conference on Computer Vision. Springer, 2016, pp. 21–37.

[4] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "MobileNets: Efficient convolutional neural networks for mobile vision applications," arXiv preprint arXiv:1704.04861, 2017.

[5] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," arXiv preprint arXiv:1409.1556, 2014.

[6] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," University of Toronto, Tech. Rep., 2009.

[7] B. Yao, X. Yang, and S.-C. Zhu, "Introduction to a large-scale general purpose ground truth database: methodology, annotation tool and benchmarks," in International Workshop on Energy Minimization Methods in Computer Vision and Pattern Recognition. Springer, 2007, pp. 169–183.

[8] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in IEEE Conference on Computer Vision and Pattern Recognition, 2009, pp. 248–255.

[9] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Doll´ar, and C. L. Zitnick, "Microsoft COCO: Common objects in context," in European Conference on Computer Vision. Springer, 2014, pp. 740–755.

[10] A. S. Razavian, H. Azizpour, J. Sullivan, and S. Carlsson, "CNN features off-the-shelf: an astounding baseline for recognition," in IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), 2014, pp. 512–519.

[11] B. C. Russell, A. Torralba, K. P. Murphy, and W. T. Freeman, "LabelMe: a database and web-based tool for image annotation," International Journal of Computer Vision, vol. 77, no. 1, pp. 157–173, 2008.

[12] Y. Yao, J. Zhang, F. Shen, X. Hua, J. Xu, and Z. Tang, "A new web-supervised method for image dataset constructions," Neurocomputing, vol. 236, pp. 23–31, 2017.

[13] A. Tegen, R. Weegar, L. Hammarlund, M. Oskarsson, F. Jiang, ˚om, "Image segmentation and D. Medved, P. Nugues, and K. Astr¨labeling using free-form semantic annotation," in IEEE International Conference on Pattern Recognition, 2014, pp. 2281–2286.

[14] S. Alabachi and G. Sukthankar, "Intelligently assisting human-guided quadcopter photography," in Proceedings of Florida Artificial Intelligence Research Society, Melbourne, FL, May 2018.

[15] D. S. Bolme, J. R. Beveridge, B. A. Draper, and Y. M. Lui, "Visual object tracking using adaptive correlation filters," in IEEE Conference on Computer Vision and Pattern Recognition, 2010, pp. 2544–2550.