CS230

Tracking Vehicles Across Multiple Viewpoints

Paul Warren
CS 230 Winter 2018
Department of Computer Science
Stanford University
pwarren@stanford.edu

Abstract

This project focuses on the problem of tracking the paths of individual vehicles across multiple surveillance cameras despite significant variations in viewpoint, scale, occlusion, illumination, and background. We frame this as a visual query system: given an image of a vehicle, query a database of vehicle images and return images that are of the same car. Existing approaches tend to have two models: an object detection model and a separate embedding model. We experiment with using just one object detection model, YOLOv3, for both tasks: we treat the vector YOLOv3 used to decide bounding boxes as the visual embeddings for the vehicle in that bounding box. Although less specialized embeddings should decrease accuracy, using only one model should significantly decrease the amount of memory and runtiem required to deploy a multi-camera tracking system, which may be an acceptable accuracy-vs-footprint tradeoff. We found that a pretrained YOLOv3 performs twice as good as random guessing and half as good as modern results, suggesting that with further training on surveillance data, YOLOv3 alone might be a promising approach.

1 Introduction

There are hundreds of millions of surveillance camera worldwide. By 2018, the Chinese government alone has installed 200 million surveillance cameras (approximately 1 camera per 7 citizens)[1]. For reference, in 2014, public and private organizations in the United States operated 40 million surveillance cameras (1 camera per 8 citizens)[2].

Some applications of this technology include monitoring traffic, optimizing logistics, tracking fugitives, and following stolen cars. The privacy and ethical issues of these applications are worthy of a report on their own. For now, we focus on the technical issues with these systems.

The biggest technical issue is resolution. Most computer vision datasets focus on images taken from handheld cameras or smartphones. However, security cameras are often aloft and have a wide field of view. The images tend to be grainy and the objects tend to be at odd angles. Additionally, there tends to be a significant difference in viewpoint, scale, occlusion, illumination, and background between different surveillance cameras.

[1] https://www.nytimes.com/2018/07/08/business/china-surveillance-technology.html

[2] https://www.nkytribune.com/2017/04/keven-moore-surveillance-cameras-are-everywhere-providing-protection-but-not-much-privacy/

CS230: Deep Learning, Winter 2018, Stanford University, CA. (LateX template borrowed from NIPS 2017.)

2 Related work

This problem is often called "vehicle re-identification". Most of this literature has not been reproduced. The most cited dataset, CompCars (published in 2015), has been cited a total of 277 times, and papers primarily published without code and often go months without citations.

What literature exists suggests three common ways of approaching this problem:

- Vehicle embeddings (like word2vec)
- Vehicle attributes (like license plate, make, model)
- Detection attributes (like where and when we detected the vehicle)

Most approaches focus on vehicle embeddings, which is what we chose to focus on. A recent paper from NVIDIA demonstrates results comparable with state-ofthe-art using triplet loss [NVIDIA]. VR-PROUD [2] claims state-of-the-art results using a CNN architecture for feature extraction followed by an unsupervised technique that enables self-paced progressive learning in addition to incorporating contextual information. Y. Lou et al. propose an end-to-end embedding adversarial learning network (EALN) capable of generating samples localized in the embedding space, finding that using these generated samples increases performance without requiring additional labeled data [EALN]. C. Wu et al. trains a vehicle feature extractor in a multi-task approach on three existing vehicle datasets and fine-tunes the feature extractor using adaptive feature learning techniques based on the space-time prior [4]. Y. Zhou and L. Shao proposes a Viewpoint-aware Attentive Multi-view Inference (VAMI) model that extracts the single-view feature for each input image and transform the features into a global multi-view feature representation so that pairwise distance metric learning can be better optimized in such a viewpoint-invariant feature space, claiming state-of-the-art results [VIEWPOINT-AWARE]

3 Dataset and Features

There are not many vehicle re-identification datasets, and most that exist are only available upon request. Below is a list of all the three datasets we downloaded and a short description of each.

- **PKU-Vehicle** [6]. 10 million crops of vehicles without annotations, meant primarily to serve as red herrings during queries to simulate real-world queries. Various locations (e.g. highways, streets, intersections, etc.), weather conditions (e.g.,sunny, rainy, foggy), illuminations (e.g., daytime and evening), shooting angles (e.g., front, side, rear) and hundreds of vehicle brands.
- **CompCars** [7]. Web and surveillence camera images. The web portion contains 136,726 images capturing entire cars and 27,618 images capturing car parts, over 163 car makes with 1,716 car models. The full car images are labeled with bounding boxes and viewpoints. Each car model is labeled with five attributes, including maximum speed, displacement, number of doors, number of seats, and type of car. The surveillance camera portion contains 50,000 car images captured in the front view.
- **VeRi** [8]. 50,000 images of 776 vehicles captured by 20 nearby cameras covering a 1 km^2 area in 24 hours labeled with varied attributes, e.g. bounding box, types, colors, and brands, license plates, spatiotemporal information. Each vehicle is captured by 2-18 cameras in different viewpoints, illuminations, resolutions, and occlusions.

We focused on the VeRi dataset, as it was the only one that had images labeled by Vehicle ID. Each vehicle was captured by multiple cameras. Each camera captured roughly 6 images per vehicle. Each image was labeled with the vehicle ID, camera ID, vehicle color, and vehicle type (e.g. sedan or bus).

VeRi breaks the images into 37,778 training images, 11,579 testing images, and 1,678 query image. During testing, you are intended to take a query image and retrieve at most N testing images, where all returned images are supposed to have the same vehicle ID. Separating the training set into a queryable set and query set is left to the researcher. Going by vehicle ID, there are no overlapping vehicles in train and test/query.

We started with a YOLOv3 [9] pretrained on MS-COCO [10]. We intended to further train YOLOv3 on VeRi images. Unfortunately, after we received the data, we discovered that the original images

Figure 1: VeRi image examples

were not included. VeRi images are just crops of vehicles. Example VeRi images are displayed in Figure 1. Because we didn't have the original frames and bounding boxes to train YOLOv3 on, we couldn't do any further training on this dataset, and did not do any preprocessing, normalization, or data augmentation.

4 Methods

We used YOLOv3 [9]. YOLOv3 is the current state-of-the-art model for real-time object detection.

YOLOv3 predicts 4 coordinates for each bounding box, t_x, t_y, t_w, t_h using k-means dimension clusters as anchor boxes. Training is performed with sum of squared error loss. YOLOv3 predicts an objectness score for each bounding box using logistic regression and the class probabilities using independent logistic classifiers. It uses binary cross-entropy loss for the class predictions.

YOLOv3 extracts features from 3 different scales using a concept similar to feature pyramid networks. From the darknet-53 feature extraction backbone, YOLOv3 adds several convolutional layers, the last of which predicts a 3-d tensor encoding bounding box, objectness, and class predictions. In YOLOv3's original COCO experiments, they predict 3 bounding boxes at each scale, so the tensor is N N [3 (4 + 1 + 80)] for the 4 bounding box offsets, 1 objectness prediction, and 80 class predictions.

For the second scale, YOLOv3 takes the feature map from the 2 previous layers and upsamples it by 2x. It also concatenates in a feature map from earlier in the network, and then adds a few more convolutional layers to process the combined feature map, and now predict a similar tensor at the second scale. This process is repeated at the third scale.

It automatically identifies and draws rectangular bounding boxes around objects of interest at a rate of roughly 15-20+ frames per second. YOLO networks divide the image into regions and predict bounding boxes and probabilities for each region.

For a 416x416 input image, YOLOv3 predicts 13x13x3 = 507 boxes for the first scale, 26x26x3 = 2028 boxes for the second scale, and 52x52x3 = 8112 boxes for the third scale, for a total of 10,647 boxes. Non-maxima suppression and IOU thresholds are then used to cut the number of boxes down significantly, often to a few or a couple dozen per image.

YOLOv3 is the third generation of the YOLO architecture. It is implemented in the darknet [11] deep learning framework, a C-based framework developed and occasionally maintained by the YOLO creator for his personal deep learning research.

There are several alternative state-of-the-art object detection models, most noticeably SSD [12], R-FCN [13], Faster-RCNN [14], and Mask-RCNN [15]. Some of these models are more accurate than YOLOv3 or return more specific results. For example, Mask-RCNN labels each individual pixel of each object ("instance segmentation") instead of drawing rough rectangular bounding boxes. This type of output would let us crop the input image down to just the car, instead of the car plus some of its surroundings, which would increase the accuracy of things like extracting the appearance vector.

However, for all of these alternatives models, the increased accuracy comes at the cost of slower speed (often single-digit frames per second compared to YOLOv3's 15-20+). For this paper, we focus on evaluating the speed and accuracy tradeoff for YOLOv3 for the problem of vehicle re-identification.

We based our model off of a PyTorch implementation of YOLOv3 [16].

5 Experiments/Results/Discussion

We ran YOLOv3 (trained on MS-COCO) with a batch size of 52 on each of the 11,579 testing images and 1,678 query images (after letterbox resizing them to 416x416). Most of the time, YOLOv3 detected exactly one object. Occasionally it detected multiple objects (in which case we chose the bounding box with the highest confidence score) or no objects (in which case we removed the image). We extract the length 255 vector YOLOv3 used to decide each bounding box and save this as an embedding for that particular image. For each query image that has an embedding, we search for the 50 most similar test images using the cosine similarity metric. Each query image is also included in the test image. Because the embedding for the images is exactly the same, this test image was always returned as the most similar result. We ignore this result and focus on the 2nd-51st closest images.

The most common metric in re-identification tasks is Cumulative Matching Characteristic (CMC). For a single query, a rank=50 query will return a vector of 50 numbers: 0 until the first image of that ID appears and 1 thereafter. For example, if an image of the the same object appears in the 3rd slot, the result would be [0, 0, 0, 1, 1...]. The CMC is this curve averaged over all queries (np.mean(res, axis=0), resulting in a length-50 vector) and plotted.

We compare our results against (1) the results described in the 2016 VeRi paper and (2) a random baseline, which returns N random images in the test set.

We found that in 691 out of 1678 cases, the YOLOv3 approach returned at least one valid response, compared to 409 positive responses for the random approach.

A visual inspection of the incorrect responses suggested at least 10% of the time, the returned response was the same color as the query car. Cars of the same type were also more common.

6 Conclusion/Future Work

We found that YOLOv3 is at least twice as good as random chance, though half as good as modern state-of-the-art-ish results. These were surprisingly promising results. YOLOv3 was trained only on objection detection on COCO (328k images) (feature vector n=255), compared to the lowest-performing result in the VeRi dataset, AlexNet's, classification task on ImageNet 1000 (1.5m) (feature vector n=4096). Further training YOLOv3 on surveillance data (e.g. CompCars) and on fine-grained

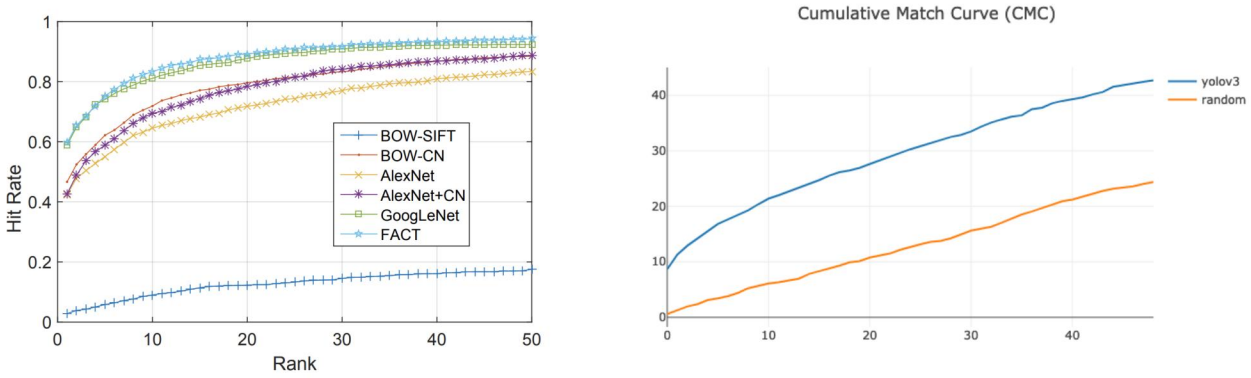

Figure 2: VeRi 2016 results (left) and our YOLOv3 vs random baseline results (right) (axes not the same scale!)

object detection (to e.g. detect different vehicle types) would likely significantly improve the quality of the results. Additionally, results may be better with more accurate but slower object detection or instance segmentation models like Faster-RCNN, Mask-RCNN, or the newly released RetinaMask [17]. Further work involves experimenting with different architectures, training on additional datasets, and evaluating using additional quantitative (like mAP and hit rate by color, vehicle type) and qualitative metrics (like easier-to-digest top 10 visualizations).

References

[1] R. Kumar, E. Weill, F. Aghdasi, and P. Sriram. "Vehicle Re-Identification: an Efficient Baseline Using Triplet Embedding". https://arxiv.org/pdf/1901.01015.pdf (2019).

[2] R.M.S. Bashir, M. Shahzad, M.M. Fraz, "VR-PROUD: Vehicle Re-identification using PROgressive Unsupervised Deep architecture," Pattern Recognition, Volume 90, Pages 52-65 (2019).

[3] Y. Lou, Y. Bai, J. Liu, S. Wang and L. Duan, "Embedding Adversarial Learning for Vehicle Re-Identification," in IEEE Transactions on Image Processing.

[4] Chih-Wei Wu, Chih-Ting Liu, Cheng-En Chiang, Wei-Chih Tu, Shao-Yi Chien; The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops, 2018, pp. 121-128

[5] Yi Zhou, Ling Shao; The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2018, pp. 6489-6498

[6] Liu, Hongye and Tian, Yonghong and Wang, Yaowei and Pang, Lu and Huang, Tiejun, "Deep Relative Distance Learning: Tell the Difference Between Similar Vehicles, 2016 Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (IEEE), pp. 2167-2175.

[7] Linjie Yang, Ping Luo, Chen Change Loy, Xiaoou Tang. A Large-Scale Car Dataset for Fine-Grained Categorization and Verification, In Computer Vision and Pattern Recognition (CVPR), 2015.

[8] X. Liu, W. Liu, H. Ma and H. Fu, "Large-scale vehicle re-identification in urban surveillance videos," 2016 IEEE International Conference on Multimedia and Expo (ICME), Seattle, WA, 2016, pp. 1-6.

[9] J. Redmon and A. Farhadi. Yolov3: An incremental improvement. arXiv preprint arXiv:1804.02767, 2018.

[10] Lin TY. et al. (2014) Microsoft COCO: Common Objects in Context. In: Fleet D., Pajdla T., Schiele B., Tuytelaars T. (eds) Computer Vision – ECCV 2014. ECCV 2014. Lecture Notes in Computer Science, vol 8693. Springer, Cham

[11] J. Redmon, "Darknet: Open source neural networks in c," http: //pjreddie.com/darknet/, 2013-2019.

[12] Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C. Y., & Berg, A. C. (2016, October). Ssd: Single shot multibox detector. In European conference on computer vision (pp. 21-37). Springer, Cham.

[13] Dai, J., Li, Y., He, K., Sun, J.: R-FCN: Object Detection via Region-based Fully Convolutional Networks. In: NIPS. pp. 379–387 (2016)

[14] Ren, S., He, K., Girshick, R., Sun, J. (2015). Faster r-cnn: Towards real-time object detection with region proposal networks. In Advances in neural information processing systems (pp. 91-99).

[15] He, K., Gkioxari, G., Dollár, P., Girshick, R. (2017). Mask r-cnn. In Proceedings of the IEEE international conference on computer vision (pp. 2961-2969).

[16] A. Kathuria, "YOLO_v3_tutorial_from_scratch," https://github.com/ayooshkathuria/YOLO_v3_tutorial_from_scratch, 2018-2019.

[17] Fu, Cheng-Yang and Shvets, Mykhailo and Berg, Alexander C. "Learning to predict masks improves state-of-the-art single-shot detection for free," arXiv preprint arXiv:1901.03353 (2019).