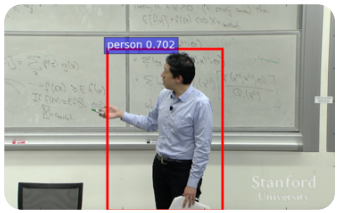
YOLOFlow: Improved Real-time Object Tracking in Video

Konstantine Buhler, John Lambert, Matthew Vilim
{buhler, johnwl, mvilim}@Stanford.edu
Professors: Andrew Ng and John Duchi TA: Rishabh Bhargava

Overview

- Given video input, we detect objects (person, car, dog, etc.).
- We implement one of the fastest image object detectors, YOLO (You Only Look Once), in TensorFlow. We call this "YOLOFlow." Further, we employ K-means clustering across images within a short, rolling window to group similar objects across frames, enabling to predict if an object is considered "similar" between frames.

These combined techniques open many possibilities, like predicting how an object is moving within the video. This full process is "Improved YOLOFlow." We apply this combination of techniques to security footage and yield some promising results. We were thrilled by what we can achieve thanks to the principles learned in this course.



Example output from our YOLO. Astoundingly, both the bounding boxes and the class types are calculated in one pass through the network.

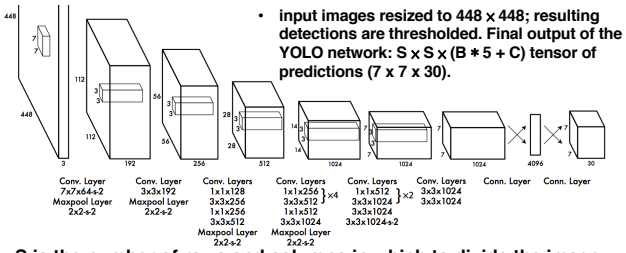
What is YOLO?

YOLO is a single feedforward neural network that predicts bounding boxes and class probabilities of an image in a single evaluation. YOLO was invented by Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi [1]. Their work builds on GoogLeNet and Network in Network [2]. We will discuss the detail in our "implementation" section.

Our YOLOFlow Implementation

Model

- 32-layer deep convolutional neural network (CNN) [1]



- input images resized to 448 x 448; resulting detections are thresholded. Final output of the YOLO network: S x S x (B * 5 + C) tensor of predictions (7 x 7 x 30).
- S is the number of rows and columns in which to divide the image.
- B is the number of objects that can be predicted in a given box .
- C is the number of classes.
- 5 terms account for the x-axis grid offset, y-axis grid offset, width, height, and confidence in each grid cell.

$$\lambda_{coord} \sum_{i=0}^{S^2} \sum_{j=0}^{B} 1_{ij}^{obj} [(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2] \quad \text{(Term 1)}$$

$$+ \lambda_{coord} \sum_{i=0}^{S^2} \sum_{j=0}^{B} 1_{ij}^{obj} [(\sqrt{w_i} - \sqrt{\hat{w}_i})^2 + (\sqrt{h_i} - \sqrt{\hat{h}_i})^2] \quad \text{(Term 2)}$$

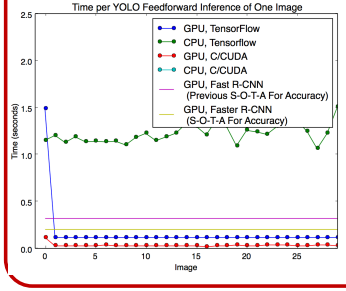
$$+ \sum_{i=0}^{S^2} \sum_{j=0}^{B} 1_{ij}^{obj} (C_i - \hat{C}_i)^2 \quad \text{(Term 3)}$$

$$+ \lambda_{noobj} \sum_{i=0}^{S^2} \sum_{j=0}^{B} 1_{ij}^{noobj} (C_i - \hat{C}_i)^2 \quad \text{(Term 4)}$$

$$+ \sum_{i=0}^{S^2} 1_{i}^{obj} \sum_{c \in classes} (p_i(c) - \hat{p}_i(c))^2 \quad \text{(Term 5)}$$

Data and Results

We consider the PASCAL VOC 2007 data set, ensuring that we achieve high accuracy when benchmarked against the results on the test set [3].
Inference: pre-trained ImageNet weights (trained for 1 week on GPU)



- 6x faster on CPU, when compared to the C implementation
- 1/3 slower on GPU than CUDA implementation, making this one of the fastest object detectors available on a CPU and a very fast GPU offering as well [5][6]

Speed Comparison of Feedforward Inference
(Averaged over 30 Image Passes)

	Time on CPU (seconds)	GPU (seconds)
Baseline YOLO CUDA/C Implementation	6.80918921676	0.0348387086452
TensorFlow YOLO Implementation	1.2247150739	0.162782869031

K-Means Extension of YOLOFlow

- Problem: YOLO detects objects within an image but has no knowledge of the similarities of frames in a video.
- We extend YOLO by providing temporal continuity in object detection.
- We use K-means clustering across images within a short, rolling window to group similar objects across frames.

- Further work could be done to improve the algorithm by using a more advanced image clustering algorithm to determine "image distance" [4].

Applying Improved YOLOFlow

- "Real world problem" -- physical security
- Several hours of video footage from the security camera of a gas station. In addition to alerting property owners to human motion, our improved algorithms could help with retailer intelligence on their customers.
- Quantify the number of customers in a store and their flow in and out
- Gas station chain interested in a partnership.

Our system has very good individual detection.

Our system does not handle some cases, like a person with a child, because of low resolution.

Sometimes predictions are too sensitive, or obfuscated objects are slightly inaccurate.

Future Work

Data Set mAP Verification: MS COCO and Lymph Node CT Lesion data sets
Directional detection: Customer Behavior and Movement Classification
GPU optimization: Further optimizing our implementation for a GPU.
Face recognition: Clustering images to determine a particular customer.

References

[1] Joseph Redmon et al. "You Only Look Once: Unified, Real-Time Object Detection". In: CoRR abs/1506.02640 (2015). URL: http : / / arxiv . org/abs/1506.02640.
[2] Min Lin, Qiang Chen, and Shuicheng Yan. "Network In Network". In: CoRR abs/1312.4400 (2013). URL: http://arxiv.org/abs/1312.4400.
[3] Everingham, M. et al. "Visual Object Classes Challenge" (VOC 2012). URL: http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html
[4] Liwei Wang et al. "On the Euclidean Distance of Images." URL: http://www.cis.pku.edu.cn/faculty/vision/wangliwei/pdf/IMED.pdf
[5] S. Ren, K. He, R.B. Girshick, and J. Sun. Faster R-CNN: towards real-time object detection with region proposal networks. 2015.
[6] Shaoqing Ren and Kaiming He and Ross Girshick and Jian Sun}, Faster {R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. NIPS, 2015.