LOW COST OBJECT DETECTION USING YOLO

Sudha Rani.J
Anurag Group of Institutions (JNTUH)

Abstract

Object tracking is one of the major fundamental challenging problems in computer vision applications due to difficulties in tracking of objects can arises due to intrinsic and extrinsic factors like deformation, camera motion, motion blur and occlusion. YOLO is innovative approach to object detection. Prior work on object detection repurposes classifiers to perform detection. Instead, frame object detection as a regression problem to spatially separated bounding boxes and associated class probabilities. A single neural network predicts bounding boxes and class probabilities directly from full images in one evaluation. Since the whole detection pipeline is a single network, it can be optimized end-to-end directly on detection performance unified architecture is extremely fast. Here base YOLO model processes images in real-time at 45 frames per second. A smaller version of the network, Fast YOLO, processes an astounding 155 frames per second while still achieving double, the map of other real-time detectors. Compared to state-of-the-art detection systems, YOLO makes more localization errors but is less likely to predict false positives on background. Finally, YOLO learns very general representations of objects. It outperforms other detection methods, including DPM and R-CNN, when generalizing from natural images to other domains like artwork.

Keywords: Occlusion, Regression, Yolo

Introduction

YOLO trains on full images and directly optimizes detection performance. This unified model has several benefits over traditional methods of object detection. First, YOLO is extremely fast. Since we frame detection as a regression problem we don't need a complex pipeline. By running our neural network on a new image at test time to predict detections. Unlike sliding window and region proposal-based techniques, YOLO sees the entire image during training and test time so it implicitly encodes contextual information about classes as well as their appearance.

Fast R-CNN, a top detection method, mistakes background patches in an image for objects because it can't see the larger context. YOLO makes less than half the number of background errors compared to Fast R-CNN.

YOLO learns generalizable representations of objects. When trained on natural images and tested on artwork, YOLO outperforms top detection methods like DPM and R-CNN by a wide margin. Since YOLO is highly generalizable it is less likely to break down when applied to new domains or unexpected inputs.

YOLO still lags state-of-the-art detection systems in accuracy. While it can quickly identify objects in images it struggles to precisely localize some objects, especially small ones. We examine these tradeoffs further in our experiments. All training and testing code is open source. Variety of pretrained models are also available to download.

Related Work:

Architecture:

Unifying the separate components of object detection into a single neural network. Our network uses features from the entire image to predict each bounding box. It also predicts all bounding boxes across all classes for an image simultaneously. This means our network reasons

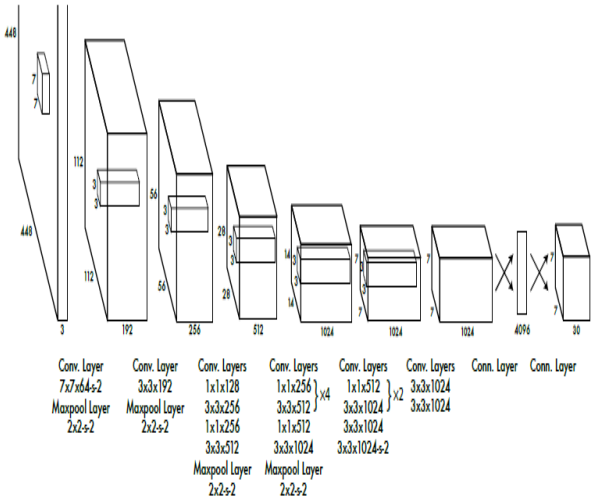
globally about the full image and all the objects in the image.

The YOLO design enables end-to-end training and real time speeds while maintaining high average precision. The system divides the input image into an S_S grid. If the center of an object falls into a grid cell, that grid cell is responsible for detecting that object. Each grid cell predicts B bounding boxes and confidence scores for those boxes. These confidence scores reflect how confident the model is that the box contains an object and how accurate it thinks the box is that it predicts. Formally we define confidence as Pr(Object) If no object exists in that cell, the confidence scores should be zero. Otherwise we want the confidence score to equal the intersection over union (IOU) between the predicted box and the ground truth. Each bounding box consists of 5

In this system model's detection as a regression problem. It divides the image into an S _S grid and for each grid cell predicts B bounding boxes, confidence for those boxes, and C class probabilities. These predictions are encoded as an S _ S _ (B _ 5 + C) tensor. For evaluating YOLO on PASCAL VOC, we use S = 7, B = 2. PASCAL VOC has 20 labelled classes so C = 20. Our final prediction is a 7 _ 7 _ 30 tensors.

Used this model as a convolutional neural network and evaluate it on the PASCAL VOC detection dataset. The initial convolutional layers of the network extract features from the image while the fully connected layers predict the output probabilities and coordinates. This network architecture is inspired by the GoogLeNet model for image classification. This network has 24 convolutional layers followed by 2 fully connected layers. Instead of the inception modules used by GoogLeNet, we simply use 1_1 reduction layers followed by 3_3 convolutional layers.

We also train a fast version of YOLO designed to push the boundaries of fast object detection. Fast YOLO uses a neural network with fewer convolutional layers (9 instead of 24) and fewer filters in those layers. Other than the size of the network, all training and testing parameters are the same between YOLO and Fast YOLO.



This detection network has 24 convolutional layers followed by 2 fully connected layers. Alternating 1 _ 1 convolutional layers reduce the features space from preceding layers. We pretrain the convolutional layers on the ImageNet classification task at half the resolution (224 _ 224 input image) and then double the resolution for detection. The final output of our network is the 7 _ 7 _ 30 tensor of predictions.

Training

By pretrain our convolutional layers on the ImageNet 1000-class competition dataset. For pretraining we use the first 20 convolutional layers followed by an average-pooling layer and a fully connected layer. We train this network for approximately a week and achieve a single crop top-5 accuracy of 88% on the ImageNet 2012 validation set, comparable to the GoogLeNet models in Caffe's Model Zoo. We use the Darknet framework for all training and inference.

We then convert the model to perform detection which show that adding both convolutional and connected layers to pretrained networks can improve performance. Following their example, added four convolutional layers and two fully connected layers with randomly initialized weights. Detection often requires fine-grained visual information so we increase the input resolution of the network from 224 _ 224 to 448 _ 448.

Our final layer predicts both class probabilities and bounding box coordinates. Normalized the bounding box width and height by the image

width and height so that they fall between 0 and 1. This parametrize the bounding box x and y coordinates to be offsets of a grid cell location so they are also bounded between 0 and 1. Linear activation function for the final layer and all other layers use the following leaky rectified linear activation:

$$\phi(x) = \begin{cases} x, & \text{if } x > 0 \\ 0.1x, & \text{otherwise} \end{cases}$$

We optimize for sum-squared error in the output of our model. Sum-squared error is used because it is easy to optimize, however it does not perfectly align with our goal of maximizing average precision. It weights localization error equally with classification error which may not be ideal. Also, in every image many grid cells do not contain any object. This pushes the "confidence" scores of those cells towards zero, often overpowering the gradient from cells that do contain objects. This can lead to model instability, causing training to diverge early on.

YOLO predicts multiple bounding boxes per grid cell. At training time, bounding box is predictor to be responsible for each object. This assign one predictor to be "responsible" for predicting an object based on which prediction has the highest current IOU with the ground truth. This leads to specialization between the bounding box predictors. Each predictor gets better at predicting certain sizes, aspect ratios, or classes of object, improving overall recall.

$$\begin{aligned}
&\lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 \right] \\
&+ \lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[\left(\sqrt{w_i} - \sqrt{\hat{w}_i} \right)^2 + \left(\sqrt{h_i} - \sqrt{\hat{h}_i} \right)^2 \right] \\
&+ \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left(C_i - \hat{C}_i \right)^2 \\
&+ \lambda_{\text{noobj}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{noobj}} \left(C_i - \hat{C}_i \right)^2 \\
&+ \sum_{i=0}^{S^2} \mathbb{1}_{i}^{\text{obj}} \sum_{c \in \text{classes}} \left(p_i(c) - \hat{p}_i(c) \right)^2
\end{aligned}$$

Note that the loss function only penalizes classification error if an object is present in that grid cell (hence the conditional class probability discussed earlier). It also only penalizes bounding box coordinate error if that predictor is "responsible" for the ground truth box (i.e. has the highest IOU of any predictor in that grid cell). If we start at a high learning rate our model often diverges due to unstable gradients.

To avoid overfitting, we use dropout and extensive data augmentation. A dropout layer with rate = .5 after the first connected layer prevents co-adaptation between layers. For data augmentation, we introduce random scaling and translations of up to 20% of the original image size. We also randomly adjust the exposure and saturation of the image by up to a factor of 1:5 in the HSV color space.

Just like in training, predicting detections for a test image only requires one network evaluation. On PASCAL VOC the network predicts 98 bounding boxes per image and class probabilities for each box. YOLO is extremely fast at test time since it only requires a single network evaluation, unlike classifier-based methods.

The grid design enforces spatial diversity in the bounding box predictions. Often it is clear which grid cell an object falls in to and the network only predicts one box for each object. However, some large objects or objects near the grid design enforces spatial diversity in the bounding box predictions. Often it is clear which grid cell an object falls in to and the network only predicts one box for each object. However, some large objects or objects near the border of multiple cells can be well localized by multiple cells. Non-maximal suppression can be used to fix these multiple detections. While not critical to performance as it is for R-CNN or DPM, non-maximal suppression adds 2-3% in map.

Differentiation:

We can increase the accuracy significantly over previous attempts. Improvements include using a small convolutional filter to predict object categories and offsets in bounding box locations, using separate predictors (filters) for different aspect ratio detections, and applying these filters to multiple feature maps from the later stages of a network to perform detection at multiple scales. With these modifications—especially using multiple layers for prediction at different scales—we can achieve high-accuracy using relatively low resolution input, further increasing detection speed. While these contributions may seem small independently, we note that the resulting system improves accuracy on real-time

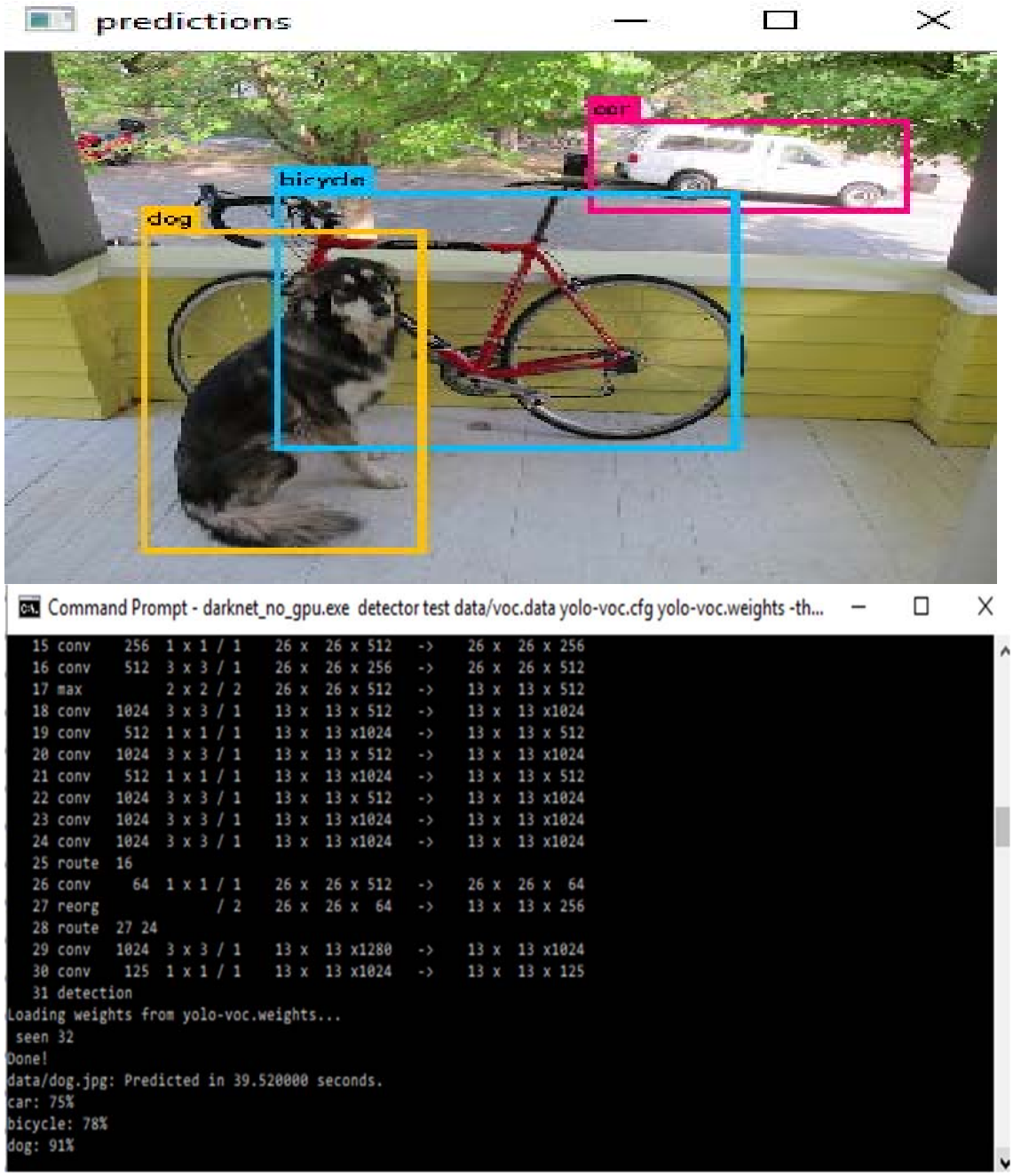
detection for PASCAL VOC from 63.4% map for YOLO to 74.3%.

Abbreviations: yolo: you only look once,
CNN: convolutional neural network
DPM: Deformable part models
VOC: visual object classes
IOU:intersection over union

Results

High speed GPU can be used to perform the operation on object detection for surveillance applications which is expensive. However low-cost object detection can be achieved by using optimum configuration of CPU (INTEL I7) which we got 39seconds of latency applied for security services such as banking, shopping malls, offices so on.

Let us consider an image listed some objects For which object car 75% weightage predefined from the data,78% for a bicycle and 91% for a dog



Conclusion

It is unified model for object detection and simple to construct and can be trained directly on full images. Unlike classifier-based approaches, YOLO is trained on a loss function that directly corresponds to detection performance and the entire model is trained jointly.

Fast YOLO is the fastest general-purpose object detector in the literature and YOLO pushes the state-of-the-art in real-time object detection. YOLO also generalizes well to new domains making it ideal for applications that rely on fast, robust object detection.

Future work

Low cost object detection model with GPU (NVIDIA GEFORCE 920MX) to be implemented with less latency

Bibliography:

1. J. Redmon. Darknet: Open source neural networks in c. http://pjreddie.com/darknet/, 2013–2016.
2. S. Gidaris and N. Komodakis. Object detection via a multiregion& semantic segmentation-aware CNN model. CoRR,abs/1505.01749, 2015
3. D. Mishkin. Models accuracy on imagenet 2012 val. https://github.com/BVLC/caffe/wiki/ Models-accuracy-on-ImageNet 2012val. Accessed: 2015-10-2
4. Fundamentals of Neural Networks: Architectures, Algorithms and Applications, 1e Paperback – 2004 by FAUSETT (Author)
5. S. Ginosar, D. Haas, T. Brown, and J. Malik. Detecting people in cubist art. In Computer Vision-ECCV 2014Workshops, pages 101–116. Springer, 2014.