Part-based auxiliary objectives with no extra labels for robust single-shot object detection

Vineeth Bhaskara[1]
bhaskara@cs.toronto.edu
Stavros Tsogkas[2]
stavros.t@samsung.com
Konstantinos G. Derpanis[2]
k.derpanis@samsung.com
Alex Levinshtein[2]
alex.lev@samsung.com

[1] Department of Computer Science, University of Toronto
[2] Samsung AI Centre Toronto

Abstract

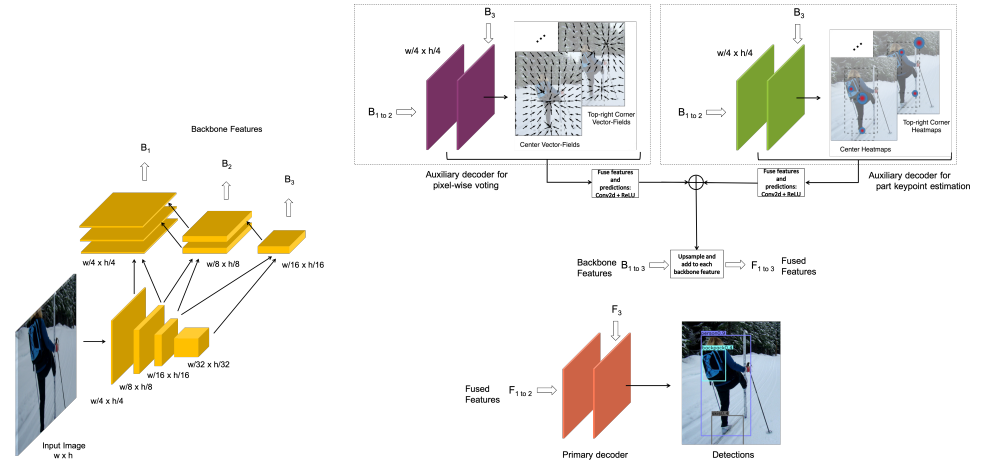
Modern single-stage detectors formulate object detection as end-to-end inference of a fully convolutional neural network without any intermediate stages. While such detectors achieve impressive speed, they often perform poorly compared to two-stage methods. One potential explanation for this is that feature maps are shared across all objects comprising a scene, impeding the learning of part-object relationships and compromising accurate object localization; in contrast to two-stage detectors that process cropped feature maps.

Inspired by self-supervised learning, we introduce auxiliary part-localization objectives that require no additional annotation and that are dataset-specific. Specifically, we employ two part-localization auxiliary tasks: i) classification of object parts, represented as class-agnostic corner and center keypoints of bounding boxes; ii) regression of class-agnostic vector fields, describing the association of objects and its parts, by having pixels "cast votes" for the relative directions of the object (or part) they belong to. We show that such auxiliary supervision improves performance of CenterNet [51] baseline, a popular single-stage detection architecture that offers an impressive balance of speed and performance.

As an additional technical contribution, we also address a discrepancy in the location of the object centers used for training and inference of the regression head in CenterNet. Our adaptations result in an improvement of $\sim$2.6 – 3.9 AP on MS COCO test-dev across various architectures, at a minor cost in run-time speed. Our best performing model with the DLA34 backbone achieves 39.4 AP at 31 FPS, and our fastest model with the ResNet-18 backbone achieves 32.2 AP at 71 FPS.

1 Introduction

Object detection is a fundamental task in computer vision that forms the basis of many real-world applications such as surveillance [12], autonomous driving [27], scene understanding [16], and text detection [1] in the wild. It also serves as a basic vision subroutine in many



(a) (b)

Figure 1: Overview of our proposed adaptation to CenterNet. (a) DLA [28] is used as the backbone for our best performing model. (b) Part-based auxiliary features and predictions are additively fused (after upsampling to appropriate resolutions) with the backbone features that are then decoded into detections.

cross-modal applications that integrate information from multiple modalities and enable exciting applications in robotics. Current state-of-the-art detection methods use deep learning to simultaneously learn a good feature representation of the image as well as localize and predict the object instances and their classes, respectively. They can generally be classified as either two-stage or single-stage detectors.

Two-stage detectors such as Faster RCNN [25] and Mask RCNN [9] involve an intermediate region proposal step. A region proposal network (RPN) realizes a sparse set of class-agnostic, rectangular region proposals that have a high likelihood of containing an object instance. Subsequently, image features are cropped and aggregated for each proposal that are, in turn, decoded by the classification and regression heads to predict the object categories and a refinement of the bounding box localization. Though such systems are trained end-to-end, during inference, multiple forward passes of the decoder heads are required as each region proposal is processed separately.

Single-stage detectors, on the other hand, forego the region proposal step and model object detection as an end-to-end inference task. An image is input to a fully-convolutional neural network that encodes it into a latent feature representation, which, in turn, is decoded directly into classification and regression heads at a resolution that is a fraction of the input image resolution. Each pixel in the output head is held responsible for detecting objects whose centers (with respect to the output head resolution) lie within it. To detect multiple objects of different scales and aspect ratios at the same output pixel, a small number of *anchor boxes* with predefined scales and aspect ratios are generally considered at each location.

Though single-stage detectors offer near real-time detection speeds, they are severely limited in terms of accuracy compared to two-stage detectors. This can be attributed to the absence of RoI cropping and aggregation that two-stage detectors are equipped with. Extracting RoI crops allows the second-stage decoder network to have a *dynamic receptive field* that is dependent on the size of the object instance in a scene. To make it more concrete,

let f^j denote the features entering the second stage (after RoI cropping and aggregation) corresponding to an object instance indexed by j. Let the feature maps in the first stage (backbone network) at layer i be denoted by f_i. Then, it is clear that the effective receptive field (with respect to f_i) of features f^{j_1} is larger than or equal to that of features f^{j_2} when the object represented by j_1 is larger than j_2. In contrast, single-stage detectors have a fixed receptive field across pixels in a given layer with respect to a given previous layer, irrespective of the sizes of the object instances comprising a scene.

One potential explanation for how this might be affecting the performance of single-stage detectors is that with a shared feature map across all object instances in a scene, learning part-object relationships implicitly becomes more challenging. That is, since each pixel has a fixed receptive field with respect to any given previous layer, features for small objects can get easily overwhelmed by background clutter or surrounding larger objects. Similarly, for very large objects, the features captured might only reflect a limited portion of the object. This makes establishing implicit part-object relationships in the latent space challenging that are, intuitively, crucial for robust object localization and detection. Two-stage detectors, on the other hand, process cropped feature maps in the second-stage that are already fairly localized.

Multi-task learning with auxiliary supervision for detection and localization of object parts is a promising direction to reduce this discrepancy of accuracy between single- and two-stage detectors. Recent single-stage detector architectures such as ExtremeNet [32] model detection as keypoint estimation, where the keypoints correspond to the four extreme points of an object along the 2D axes of the image plane. These keypoints parametrize the extreme parts of objects that are subsequently grouped using geometrical constraints to decode object bounding boxes. This method requires additional labels for the extreme points of an object in addition to the bounding box annotations, and is prohibitively expensive and time-consuming to reproduce it on custom object detection datasets. Moreover, explicitly grouping the detected parts significantly reduces the detection speed.

Inspired by self-supervised learning, in this paper, we ask the following question:
Given an object detection dataset, can auxiliary tasks be designed to localize parts and improve object detection accuracy while i) maintaining a good detection speed and ii) requiring no additional annotations?

Self-supervised methods learn an effective latent representation of data by modeling proxy tasks using neural networks where inputs and targets are both functions of unlabeled data. Recent works [3, 6, 17, 19, 20, 29] explore a variety of proxy tasks that can be formulated to learn effective representations that generalize across many applications. In this paper, we propose part-based auxiliary tasks that are conditional on a given (labeled) object detection dataset where inputs and targets are functions of both data and its annotation. We employ multi-task learning to enrich the latent representation, thereby, improving the accuracy of the primary task of object detection while having no overhead of explicitly post-processing the part-based predictions.

Specifically, we model object parts as corners and center of bounding boxes, and demonstrate two auxiliary part-based objectives, one as a classification target, and another as a regression target. Our proxy tasks include i) estimating keypoints of four categories of corners, namely, top-left, top-right, bottom-left, and bottom-right, agnostic to different object classes; and ii) regressing pixel-wise vector-fields that cast a vote to the direction of object parts relative to itself, also agnostic to the object classes. We choose our proxy tasks to be agnostic to the object classes as we note that part-based objectives are, intuitively, more crucial for accurate object localization than for improving object classification since

localization involves estimating the extent of the object instance in a scene, while accurate classification can usually be achieved just by local texture recognition [5]. We not only show that each of these tasks individually improves the detection accuracy but also demonstrate that a combination of both the tasks boosts the detection performance further.

We evaluate our part-based objectives on CenterNet [31], a recent single-stage detector that offers an impressive balance of speed and performance, that serves as a strong baseline. Figure 1 gives an overview of our part-based extension to CenterNet.

Finally, we address a discrepancy between training and inference of CenterNet where the regression head is trained at the true object center pixels but inferred at the detected center pixels. Observing that most prevalent errors in the estimated object centers have a L_∞ distance of 1 from the true center pixels, we propose a novel modification to the regression head and the loss function to further improve localization.

Our adaptations result in an improvement of $\sim 2.6 - 3.9$ AP on MS COCO *test-dev* (with a relative improvement of $\sim 7 - 13$ % over baseline) across various architectures, at a minor cost in run-time speed. Our best performing model with the DLA34 backbone achieves 39.4 AP at 31 FPS, and our fastest model with the ResNet-18 backbone achieves 32.2 AP while running at a speed of 71 FPS.

2 Related works

Anchor-based single-stage detection. Single-stage detection approaches do not employ a distinct region proposal step, but rather directly output object predictions in the dense image (or feature) grid. Deformable Parts Models (DPMs) [4], is a prominent early example of a part-based single-stage detector in the pre-deep learning era. DPMs learn a set of HOG templates [2] modeling objects and their parts at different viewpoints, using an latent-SVM classifier. These templates are then convolved with the image in a sliding window fashion, looking for high-confidence matches. DPM variants dominated object detection until the impact of feature learning became apparent [7, 26]. Motivated by such early part-based methods, in this paper, we probe if explicit part-based supervision improves deep learning-based object detection methods.

One of the earliest deep learning-based single-stage detection method with near real-time speeds is YOLO (You Only Look Once) [24] by Redmon *et al*. They spatially divide the image into a 7×7 grid, in which each cell is "responsible" for up to two object proposals, predicting objectness (whether an object is present) and class confidence scores, as well as bounding box coordinates and size (height, width). The entire framework is trained in an end-to-end manner. However, using a coarse grid and limiting the number of detections per grid cell compromises recall, especially in the case of small or occluded objects.

These limitations were later addressed in extensions to YOLO [22], as well as in the SSD framework [15], with the introduction of *anchor boxes*. Increasing the size of the anchor set potentially improves recall but also increases training and inference time, so picking good anchor priors is important. YOLOv2 [22] uses k-means clustering on the ground truth boxes of the training set to select anchor priors, while RefineDet [30] introduces an anchor refinement module (ARM), that removes unpromising anchors and refines the ones that survive.

Addressing the limitation of severe class imbalance between positive and negative anchors in single-stage detectors, Lin *et al*. [13] introduced focal loss to suppress the gradients of easy negative examples instead of discarding them that has shown to be markedly more effective than naive hard negative mining.

YOLOv3 [23], a revision of YOLOv2 [22] with incremental improvements such as a deeper backbone architecture, independent object classifiers in the place of softmax predictions, detecting objects at multiple scales, offered the best trade-off between speed and accuracy until a recent work by Zhou *et al*. [31] who introduced CenterNet.

CenterNet [31] builds on the previous works and simplifies the decoder architecture. They propose enlarging the output resolution to a quarter of the input image resolution instead of having multiple anchors. Thus, featuring a fine output grid where each pixel can detect up to a single instance of a particular object class. They employ focal loss, and show that non-maximal suppression (NMS) is unnecessary and instead simply extract local peaks in the object classification head to detect bounding box centers. Simiar to previous methods, they regress the size (height, width) by a separate regression head that is kept class-agnostic.

Such simplifications result in an improvement over YOLOv3 while still running twice as fast. We, therefore, use CenterNet as our baseline model and propose improving the accuracy further using part-based objectives.

Keypoint-based single-stage detection. Keypoint-based detectors are a special subclass of single-stage detectors, that frame object detection as a keypoint estimation task. The model outputs a heatmap of "keypoints" that can lie on a bounding box [11] or an object itself [32]. The keypoint locations are parametrized as local peaks (generally, in a 3×3 window) of the heatmap. These keypoints are then explicitly grouped to define the bounding box.

One can draw conceptual similarities between keypoint-based methods and part-based modeling by interpreting object keypoints as "parts". Many pose estimation works refer the terms "anatomical keypoints", "joints", and "parts" interchangeably. Therefore, part-based objectives may be framed in terms of keypoints.

CornerNet [11] learns to predict the locations of the top-left and bottom-right corners ("keypoints") of the object bounding box for each object class in the dataset. It also estimates an associative embedding [18] at each of the detected keypoint locations that are then used to group them together and fully define the object bounding box. Predicting keypoints for each possible object class scales the output head multiplicatively with the number of keypoint types k ($k = 2$ for CornerNet). Moreover, post-processing involved in grouping the detected keypoints further affects the detection speed. In contrast, we choose class-agnostic estimation of $k = 4$ keypoints (corresponding to the corners of bounding boxes) as one of our part-based auxiliary objectives.

ExtremeNet [32], on the other hand, uses four extreme points on the object boundary instead of the bounding box corners for keypoint estimation, and, therefore requires additional annotation. In contrast, our part-localization objectives require no additional labels.

3 Technical approach

In the following sections, we start by reviewing CenterNet that serves as our baseline model, then we introduce notation, and describe our proposed extension in more detail. Throughout the text, whenever we refer to pixel coordinates or offsets, these are with respect to the resolution of the output feature map of the network, unless otherwise specified. Also, note that upper case letters with a hat denote predicted outputs of the network while the corresponding lower case letters denote the ground truth values (unless explicitly mentioned otherwise).

3.1 CenterNet: Overview

CenterNet takes a $W \times H \times 3$ sRGB image, I, as input, and outputs a class-specific heatmap $\hat{\mathbf{Y}} \in [0,1]^{\frac{W}{R} \times \frac{H}{R} \times C}$, where W, H are the image width and height, respectively, $R = 4$ is the downsampling factor between the input image and the output heatmap, and C is the number of object categories. Ideally, $\hat{Y}_{x,y,c} = 1$ when the center of an object belonging to category c falls inside the grid cell denoted by (x,y), and $= 0$ otherwise. To recover from discretization errors, introduced from downsampling, the network also predicts $\hat{\mathbf{O}}$, a class-agnostic $\frac{W}{R} \times \frac{H}{R} \times 2$ map of offsets between the discretized and the absolute true center coordinates in the output heatmap. Finally, the network outputs another class-agnostic $\frac{W}{R} \times \frac{H}{R} \times 2$ map, $\hat{\mathbf{S}}$, of height and width predictions that are used to fully define an object bounding box, given its center.

The regression output heads, namely, the size and offset heads, are trained to minimize the L1 loss computed at the ground truth (GT) center pixels of object bounding boxes comprising an image. The respective loss terms (per image) can be written as

$$\mathcal{L}_{\text{size}} = \frac{1}{N}\sum_{i=1}^{N}\left\|\hat{\mathbf{S}}_{\lfloor \mathbf{p}_i \rfloor} - \mathbf{s}_i\right\|_1 = \frac{1}{N}\sum_{i=1}^{N}\left|\hat{H}_{\lfloor \mathbf{p}_i \rfloor} - h_i\right| + \frac{1}{N}\sum_{i=1}^{N}\left|\hat{W}_{\lfloor \mathbf{p}_i \rfloor} - w_i\right|, \text{ and}$$

$$\mathcal{L}_{\text{off}} = \frac{1}{N}\sum_{i=1}^{N}\left\|\hat{\mathbf{O}}_{\lfloor \mathbf{p}_i \rfloor} - (\mathbf{p}_i - \lfloor \mathbf{p}_i \rfloor)\right\|_1, \tag{1}$$

where i indexes an object instance, $\mathbf{p}_i$ is the vector of real valued coordinates of the bounding box center in the output resolution, $\lfloor \mathbf{p}_i \rfloor$ is the discretized pixel coordinates, and $\mathbf{s}_i = (w_i, h_i)$ is the ground truth size (height and width) of the object i. The classification head is trained with penalized focal loss (normalized by the number of objects), treating the center pixels as positives and the rest as negatives. This is given by

$$\mathcal{L}_{\text{class}} = \frac{-1}{N}\sum_{x,y,c}\begin{cases}(1-\hat{Y}_{x,y,c})^{\alpha}\log(\hat{Y}_{x,y,c}) & \text{at center pixels,}\\ (1-Y_{x,y,c})^{\beta}(\hat{Y}_{x,y,c})^{\alpha} & \\ \log(1-\hat{Y}_{x,y,c}) & \text{otherwise,}\end{cases} \tag{2}$$

where $Y_{x,y,c} = \max_{i \in c} \exp\left(-\frac{(x-\lfloor \mathbf{p}_i \rfloor_x)^2 + (y-\lfloor \mathbf{p}_i \rfloor_y)^2}{2\sigma_i^2}\right)$ is the maximum value at (x,y) among Gaussians centered at center pixel coordinates $\lfloor \mathbf{p}_i \rfloor$ of objects i that belong to class c. The variance σ_i^2 is dependent on the bounding box size, following CornerNet [11], to smoothly penalize pixels farther away from the object centers. The final, combined loss function is

$$\mathcal{L} = \mathcal{L}_{\text{class}} + \lambda_{\text{off}}\,\mathcal{L}_{\text{off}} + \lambda_{\text{size}}\,\mathcal{L}_{\text{size}}, \tag{3}$$

with $\lambda_{\text{off}} = 1.0$, $\lambda_{\text{size}} = 0.1$, $\alpha = 2$, and $\beta = 4$.

During inference, CenterNet finds objects in an image by i) extracting peak locations (within a 3×3 neighborhood) $\{(\hat{x}_i, \hat{y}_i)\}$ for each object class from the predicted center heatmaps $\hat{\mathbf{Y}}$; ii) refining them using the predicted offsets $\{(\delta\hat{x}_i, \delta\hat{y}_i)\} \in \hat{\mathbf{O}}$; iii) retrieving the width $\{\hat{w}_i\}$ and height $\{\hat{h}_i\}$ of the corresponding boxes from $\hat{\mathbf{S}}$ at the detected center pixels. The coordinates of the top-left and the bottom-right corners of bounding box indexed by i are simply inferred as

$$(c_x^{tl}, c_y^{tl})_i = (\hat{x}_i + \delta\hat{x}_i - \frac{\hat{w}_i}{2}, \hat{y}_i + \delta\hat{y}_i - \frac{\hat{h}_i}{2}), \text{ and } (c_x^{br}, c_y^{br})_i = (\hat{x}_i + \delta\hat{x}_i + \frac{\hat{w}_i}{2}, \hat{y}_i + \delta\hat{y}_i + \frac{\hat{h}_i}{2}),$$

respectively. For notational convenience, in the rest of the text we refer to the decoder network responsible for predicting the tuple $\{\hat{\mathbf{Y}}, \hat{\mathbf{O}}, \hat{\mathbf{S}}\}$ as the "detector branch", and refer $\hat{\mathbf{Y}}$ and $\{\hat{\mathbf{O}}, \hat{\mathbf{S}}\}$ as the "classification head" and "regression head", respectively.

3.2 Part-based extension to CenterNet

3.2.1 Corner keypoints as object parts

We propose augmenting CenterNet with an additional decoder branch that estimates part-based corner keypoints across object classes. We consider detecting regions of the image for four classes of corners, namely, $k \in \{\text{top-left, top-right, bottom-left and bottom-right}\}$. We compute class-agnostic heatmaps $\hat{\mathbf{Y}}_{\text{corners}} \in [0,1]^{\frac{W}{R} \times \frac{H}{R} \times 4}$ for different corner keypoints in an identical way to the center heatmaps of the detector branch.

Ideally, $\hat{Y}_{\text{corners}}|_{x,y,k} = 1$ when an object corner (irrespective of the object class) of type k falls inside the grid cell denoted by (x,y). The auxiliary head is trained using the penalized focal loss in Eq. (2), i.e.,

$$\mathcal{L}_{\text{corners}} = \frac{-1}{4N} \sum_{x,y,k} \begin{cases} (1-\hat{Y}_{\text{corners}}|_{x,y,k})^{\alpha} \log(\hat{Y}_{\text{corners}}|_{x,y,k}) & \text{at corner pixels,} \\ (1-Y_{\text{corners}}|_{x,y,k})^{\beta} (\hat{Y}_{\text{corners}}|_{x,y,k})^{\alpha} & \\ \log(1-\hat{Y}_{\text{corners}}|_{x,y,k}) & \text{otherwise,} \end{cases} \tag{4}$$

where $Y_{\text{corners}}|_{x,y,k} = \max_i \exp\left(-\frac{(x-\lfloor \mathbf{p}_i^k \rfloor_x)^2 + (y-\lfloor \mathbf{p}_i^k \rfloor_y)^2}{2\sigma_i^2}\right)$ is the maximum value at (x,y) among Gaussians centered at corner pixel coordinates $\lfloor \mathbf{p}_i^k \rfloor$ of corner type k across all objects indexed by i. The variance σ_i^2 is dependent on the bounding box size as before. We use identical hyperparameters ($\alpha = 2$, $\beta = 4$) as used for the classification head of CenterNet.

3.2.2 Pixel-wise vector-field casting votes for object parts

In this section, we propose a part-based regression task where each pixel in the output head predicts a unit vector that "casts a vote" for the likely locations of object parts (parametrized as corners and center of bounding boxes). The pixel-wise predictions estimate regions of the image plane where object parts might occur based on a local context around the individual pixels. Such a training objective, intuitively, encourages latent representations that make context around parts more discriminative and informative.

We propose an additional decoder branch that estimates the pixel-wise vector-field $\hat{\mathbf{V}}^i$ where each pixel casts vote for the relative direction of an object part of type i (where i can be one of four corners or center of bounding boxes) likely occurring. We represent the vector field by two channels denoting the x and y components of the unit vector predicted at each pixel. That is, $\hat{\mathbf{V}}^i$ is of dimensions $\frac{W}{R} \times \frac{H}{R} \times 2$.

Determining the ground truth vector-field when a scene contains a single object is straightforward. Each pixel should ideally be voting for the keypoint of a unique object that is present in the image. PVNet [21] uses a similar representation and a RANSAC-based voting scheme to estimate a single keypoint that lies within or outside the image plane. Unlike the simple case, object detection on general scenes has to deal with multiple, potentially overlapping, objects. In such cases, which one of the keypoints should each pixel in $\hat{\mathbf{V}}^i$ vote for? We resolve this ground truth assignment ambiguity after establishing notation.

For convenience of notation, let $\mathbf{k}_j^i = (k_{jx}^i, k_{jy}^i)$ represent a keypoint of object j and type i, and w_j and h_j denote the width and height of object j. Let $B(\mathbf{x}, w_j, h_j)$ represent a rectangular region that is centered at $\mathbf{x}$ with width and height (w_j, h_j), respectively. Let $\mathbf{p}$ denote an arbitrary pixel coordinate $(p_x, p_y) \in \hat{\mathbf{V}}^i$.

We assume the context around a given keypoint scales with the size of the object to which the keypoint belongs to. Therefore, we only consider pixels within a region enclosed by a hypothetical rectangular box $B(\mathbf{k}_j^i, w_j, h_j)$ to carry enough context for the keypoint $\mathbf{k}_j^i$. Pixels $\mathbf{p} \notin \bigcup_l B(\mathbf{k}_l^i, w_l, h_l)$ are not assigned any ground truth unit vector and are masked out during training since such regions, far away from any object, carry weaker context. Pixels $\mathbf{p} \in \bigcap_l B(k_{lx}^i, k_{ly}^i, w_l, h_l)$ that belong to an overlapping region of context for more than one keypoint are assigned ground truth votes based on a "affinity" score $d_{\mathbf{k}}(\mathbf{p})$ of a pixel $\mathbf{p}$ to a keypoint $\mathbf{k}$, that we define subsequently.

We summarize the ground truth assignment of votes to pixels $(p_x, p_y) \in \hat{\mathbf{V}}^i$ for keypoint of type i as follows:

$$\hat{\mathbf{V}}_{\mathbf{p}}^i = \begin{cases} \text{masked from training} & \text{if } \mathbf{p} \notin \bigcup_l B(\mathbf{k}_l^i, w_l, h_l), \\ \dfrac{\lfloor \mathbf{k}_j^i \rfloor - \mathbf{p}}{\| \lfloor \mathbf{k}_j^i \rfloor - \mathbf{p} \|_2 + \varepsilon} & \text{else if } \mathbf{p} \in B(\mathbf{k}_j^i, w_j, h_j) \text{ and } \mathbf{p} \notin B(\mathbf{k}_l^i, w_l, h_l) \forall l \neq j, \\ \dfrac{\lfloor \operatorname{argmax}_{\mathbf{k} \in \{\mathbf{k}_l^i\}_l} d_{\mathbf{k}}(\mathbf{p}) \rfloor - \mathbf{p}}{\| \lfloor \operatorname{argmax}_{\mathbf{k} \in \{\mathbf{k}_l^i\}_l} d_{\mathbf{k}}(\mathbf{p}) \rfloor - \mathbf{p} \|_2 + \varepsilon} & \text{otherwise,} \end{cases} \tag{5}$$

where $\{\mathbf{k}_l^i\}_l$ denotes the set of all keypoints of type i across objects present in the image, and the affinity score $d_{\mathbf{k}}(\mathbf{p})$ is defined as

$$d_{\mathbf{k}_j^i}(\mathbf{p}) = \max_{w',\, h'} \text{IoU}\{B(\mathbf{p}, w', h'), B(\mathbf{k}_j^i, w_j, h_j)\}, \tag{6}$$

that represents the maximum IoU (intersection-over-union) score between an arbitrary rectangular box centered at $\mathbf{p}$ and the box $B(\mathbf{k}_j^i, w_j, h_j)$. (See supplementary material for more details).

We train the vector-field outputs to minimize the L1 regression loss given by

$$\mathcal{L}_{\text{vectors}}^i = \sum_{\mathbf{p} \in \bigcup_l B(\mathbf{k}_l^i, w_l, h_l)} \frac{1}{A_{\mathbf{p}}} \left\| \hat{\mathbf{V}}_{\mathbf{p}}^i - \mathbf{v}_{\mathbf{p}}^i \right\|_1, \tag{7}$$

where $\mathbf{v}_{\mathbf{p}}^i$ are the ground truth votes and $A_{\mathbf{p}}$ represents the area of the object bounding box whose keypoint the pixel $\mathbf{p}$ votes for.

3.2.3 Fusing auxiliary features and predictions into the primary decoder network

We fuse the auxiliary predictions and features into the detector branch additively. Specifically, we concatenate the auxiliary predictions and the features preceding it, apply a 3×3 convolution, followed by a ReLU non-linearity. These transformed part-based features are then added to the features of the detector branch at various scales after bilinear upsampling to make the dimensions commensurate. More details on the fusion architecture across various backbones can be found in the supplementary material.

Figure 2: Distribution of L_∞ error between predicted center pixels and the ground truth center pixels, for CenterNet on the MS COCO *val* dataset.

3.3 Repurposing the regression head

CenterNet decodes bounding boxes as rectangular boxes centered at the predicted center keypoint pixels with dimensions (width, height) that are inferred from the regression branch (at the predicted center pixels). But during training, the center keypoints and the size regression outputs are trained only at the true center pixels. Therefore, an erroneous center keypoint prediction, retrieves an imperfect size estimate, compounding the overall localization error of the detections further.

Figure 2 analyzes the distribution of center keypoint localization errors in the baseline CenterNet model. Evidently, about 35% of the errors are single-pixel displacement errors from the true centers (L_∞ error of 1).

We modify the regression head of CenterNet such that the network is no longer constrained to decode detections as centered boxes about the predicted (and possibly erroneous) center estimates. Specifically, we repurpose the regression head $\{\hat{\mathbf{O}}, \hat{\mathbf{S}}\}$ to predict $\hat{\mathbf{E}}$, a class-agnostic $\frac{W}{R} \times \frac{H}{R} \times 4$ map of offsets from a candidate center pixel to the four edges of the bounding box.

We train pixels around the true centers within a 3×3 window to predict the offsets to the four edges of the ground truth bounding box. This enables the regression head to compensate for single-pixel errors in the center estimates by predicting appropriate offsets to the edges such that the detections can be fully recovered.

Since detections are decoded at the local peaks of the center heatmap $\hat{\mathbf{Y}}$ during inference, we might want to emphasize gradients at such local peaks during training as well. Therefore, we weight the gradients to the regression head at each pixel $\mathbf{p}$ by the maximum probability $\hat{Y}^{max}_{\mathbf{p}} = \max_c \hat{Y}_{p_x,p_y,c}$ of it being a center keypoint across object classes (note that we don't track gradients for the weighting term back to the keypoint outputs). This emphasizes training the pixels that are local peaks in the center heatmap. We compute the following loss for training the offsets to the edges

$$\mathcal{L}_{\text{edge offsets}} = \frac{1}{N} \sum_{i=1}^{N} \left\{ \frac{\sum_{\mathbf{p} \in \mathrm{W}_{3\times3}(\mathbf{p}_i)} \hat{Y}^{max}_{\mathbf{p}} \cdot \left\| \hat{\mathbf{E}}_{\mathbf{p}} - \mathbf{e}_{\mathbf{p}} \right\|_1}{\sum_{\mathbf{p} \in \mathrm{W}_{3\times3}(\mathbf{p}_i)} \hat{Y}^{max}_{\mathbf{p}} + \varepsilon} \right\} \tag{8}$$

where N is the number of objects in the image, $\mathrm{W}_{3\times3}(\mathbf{p}_i)$ denotes a 3×3 window of pixels around the true center $\mathbf{p}_i$ of object i and $\mathbf{e}_{\mathbf{p}}$ is the vector of true offsets to the edges.

Given the predicted offsets $\hat{\mathbf{E}}_{\mathbf{p}} = (\hat{w}_l, \hat{w}_r, \hat{h}_t, \hat{h}_b)$ to the edges at a local peak $\hat{\mathbf{x}} = (\hat{x}, \hat{y})$ of the center heatmap, we decode the coordinates of the top-left and the bottom-right corners

of the inferred bounding box as

$$(c_x^{tl}, c_y^{tl}) = (\hat{x} - \hat{w}_l,\ \hat{y} - \hat{h}_t), \text{ and } (c_x^{br}, c_y^{br}) = (\hat{x} + \hat{w}_r,\ \hat{y} + \hat{h}_b).$$

4 Experiments

We evaluate our model on the MS COCO [12] dataset, using the DLA [28] and ResNet [8] backbone architectures. Figure 1 sketches an overview of our adaptations to CenterNet with the DLA backbone. A detailed illustration of the baseline model architectures and our adaptations are presented in the supplementary material. Our final model with both the part-based auxiliary tasks achieves the best performance. The complete training objective is given by

$$\mathcal{L} = \underbrace{\mathcal{L}_{\text{class}} + \lambda\ \mathcal{L}_{\text{edge offsets}}}_{\mathcal{L}_{\text{detector}}} + \underbrace{\alpha\ \mathcal{L}_{\text{corners}} + \beta\ \mathcal{L}_{\text{vectors}}}_{\mathcal{L}_{\text{auxiliary}}}, \tag{9}$$

where $\mathcal{L}_{\text{vectors}} = \sum_i \mathcal{L}^i_{\text{vectors}}$ and $\lambda = 0.1$, $\alpha = 0.3$, $\beta = 0.035$.

All the networks are trained on images with a resolution of 512×512, after resizing and padding to preserve the aspect ratio. Following [31], we use random flip, random scaling and cropping, and color jittering as data augmentations. We use the ADAM optimizer [10] with a batch size of 128 at a learning rate of 5e-4. The training schedules are not tuned and left at the defaults used by the baseline CenterNet model.

4.1 Results

Table 1 shows our ablation studies on COCO *val* with the DLA architecture trained for 140 epochs (1x). We perform inference in the original image resolution on a single NVIDIA P100 GPU. Augmenting CenterNet with part-level keypoint estimation improves AP from 36.6 to 37.8 (+1.2 points) with a minor impact on the inference speed. Similarly, the vector-fields result in an improvement of 1.4 AP over the baseline (from 36.6 to 38.0). Using both together results in a total improvement of $\sim$ 2.4 AP, indicating that the two representations carry complementary information. Finally, using our edge-offset representation for regressing the box sizes, adds a further improvement of 0.2 AP.

To disentangle performance gain due to the larger number of parameters given the two additional auxiliary decoders, we rerun our part-based CenterNet model (without the repurposed regression head) on DLA1x backbone with α and β set to zero (i.e., no auxiliary supervision through the loss). We observe 37.8 AP on COCO val which is about 1.2 AP lower than the case when explicit part-based supervision is provided.

Table 2 compares our final model with the baseline on COCO *test-dev*. Our model improves AP by 2.6, 3.2, and 3.9 points on DLA34, ResNet-101, and ResNet-18 backbone architectures, respectively, using *no augmentations* at test-time. Improvements are consistent for different IoU thresholds, object sizes, and test-time augmentation schemes, and are more pronounced at high IoU thresholds (AP_{75}), verifying our model's effectiveness in reducing localization errors. Our best model with the DLA34 backbone achieves 39.4 AP running at 31 FPS and our fastest model with the ResNet-18 backbone scores 32.2 AP while running at an impressive speed of 71 FPS.

Qualitative results of our detections and part-based auxiliary predictions as well as evaluations on Pascal VOC detection dataset are available in the supplementary material.

Auxiliary Tasks		Sizes Encoded	*AP*			AP_{50}			AP_{75}			FPS		
Corners	Vectors	as Edge Offsets	N.A.	F	MS	N.A.	F	MS	N.A.	F	MS	N.A.	F	MS
✗	✗	✗	36.6	38.5	41.2	54.1	56.4	59.8	39.7	42.1	44.6	45.4	25	4.6
✓	✗	✗	37.8	40.0	42.3	54.6	57.0	60.3	40.8	43.6	45.6	35.7	20.4	3.7
✗	✓	✗	38.0	39.7	42.1	55.2	56.9	60.0	41.4	43.3	45.5	35.7	20.4	3.7
✗	✗	✓	36.8	39.1	41.2	54.0	56.4	59.5	39.3	41.9	44.4	45.4	25	4.6
✓	✓	✗	39.0	40.5	42.8	55.8	57.2	**60.5**	**42.1**	43.9	**46.1**	31.3	17.0	3.2
✓	✓	✓	**39.2**	**40.8**	**43.0**	**56.0**	**57.5**	60.4	**42.1**	**44.0**	**46.1**	28.0	17.5	3.2

Table 1: Ablation study of our adaptations on CenterNet with the COCO validation set. The DLA backbone model is trained for 1x schedule: without test augmentation (N.A.), with flip testing (F), and with multi-scale augmentation (MS).

Backbone	Init	Model	*AP*	AP_{50}	AP_{75}	AP_S	AP_M	AP_L	FPS
DLA34-2x	ImageNet	CenterNet	36.8 / 39.0 / 41.5	54.4 / 56.8 / 60.2	40.2 / 42.5 / 44.9	18.8 / 20.2 / 21.9	41.0 / 42.6 / 43.4	48.1 / 50.9 / 55.8	43.5 / 25.0 / 4.5
		Ours	**39.4 / 41.2 / 43.6**	**56.1 / 57.7 / 60.9**	**42.1 / 44.3 / 46.8**	**19.6 / 21.0 / 23.3**	**42.5 / 44.3 / 45.8**	**52.3 / 54.0 / 58.0**	31.3 / 17.2 / 3.1
DLA34-1x	ImageNet	CenterNet	36.5 / 38.5 / 41.2	54.3 / 56.5 / 60.0	39.6 / 41.8 / 44.6	18.1 / 19.3 / 21.3	40.8 / 42.4 / 43.1	48.1 / 50.7 / 55.6	43.5 / 25.0 / 4.5
		Ours	**39.2 / 40.9 / 43.2**	**56.1 / 57.6 / 60.8**	**42.1 / 44.1 / 46.4**	**19.9 / 21.3 / 23.3**	**42.3 / 43.9 / 45.2**	**51.6 / 53.2 / 57.1**	31.3 / 17.0 / 3.2
ResNet-101	ImageNet	CenterNet	34.5 / 36.1 / 39.5	53.3 / 54.9 / 59.0	36.8 / 38.6 / 42.5	14.1 / 15.3 / 19.0	38.9 / 40.7 / 42.0	48.6 / 50.4 / 55.0	33.3 / 21.7 / 4.0
		Ours	**37.7 / 39.4 / 42.6**	**54.9 / 56.6 / 60.6**	**40.2 / 42.2 / 45.8**	**16.7 / 18.2 / 22.2**	**41.9 / 43.6 / 45.2**	**52.6 / 54.2 / 57.7**	27.0 / 16.7 / 3.1
ResNet-18	ImageNet	CenterNet	28.3 / 30.1 / 33.6	45.9 / 48.0 / 52.4	29.9 / 32.0 / 35.9	9.7 / 10.8 / 14.6	31.2 / 33.1 / 34.6	41.1 / 43.4 / 47.8	111 / 71.4 / 14
		Ours	**32.2 / 34.0 / 37.0**	**48.5 / 50.3 / 54.4**	**34.1 / 36.1 / 39.5**	**12.1 / 13.5 / 17.3**	**34.6 / 36.6 / 38.2**	**45.9 / 47.9 / 51.3**	71.4 / 43.5 / 7.8

Table 2: Performance of our final models on MS COCO *test-dev* (with auxiliary tasks and edge offsets for decoding box sizes).

5 Conclusion

In this work, we propose modeling part-based auxiliary tasks on a given object detection dataset without requiring additional annotations. We demonstrate two such part-based proxy tasks, one as a classification problem of detecting corner keypoints across objects of any class, and another as a regression problem of inferring pixel-wise "votes" for the relative directions of object parts based on local context around individual pixels.

We show that augmenting such part-based tasks as auxiliary supervision in a multi-task learning setting, improves the accuracy on the primary task of object detection for CenterNet, a strong baseline single-stage detector that offers an impressive balance of speed and accuracy. Finally, we repurpose the regression head of CenterNet to help the network be robust to the most prevalent type of localization errors. We propose weighting the regression loss with the center keypoint probability to emphasize training pixels that are local peaks of the heatmap to address the discrepency between training and inference phases.

Acknowledgements

We would like to thank Prof. Roger Grosse and Prof. Allan Jepson for their invaluable feedback and continuous support throughout the project.

References

[1] Youngmin Baek, Bado Lee, Dongyoon Han, Sangdoo Yun, and Hwalsuk Lee. Character region awareness for text detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9365–9374, 2019.

[2] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. 2005.

[3] Carl Doersch, Abhinav Gupta, and Alexei A Efros. Unsupervised visual representation learning by context prediction. In *Proceedings of the IEEE international conference on computer vision*, pages 1422–1430, 2015.

[4] Pedro F Felzenszwalb, Ross B Girshick, David McAllester, and Deva Ramanan. Object detection with discriminatively trained part-based models. *IEEE transactions on pattern analysis and machine intelligence*, 32(9):1627–1645, 2009.

[5] Robert Geirhos, Patricia Rubisch, Claudio Michaelis, Matthias Bethge, Felix A Wichmann, and Wieland Brendel. Imagenet-trained cnns are biased towards texture; increasing shape bias improves accuracy and robustness. *arXiv preprint arXiv:1811.12231*, 2018.

[6] Spyros Gidaris, Praveer Singh, and Nikos Komodakis. Unsupervised representation learning by predicting image rotations. *arXiv preprint arXiv:1803.07728*, 2018.

[7] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 580–587, 2014.

[8] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[9] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017.

[10] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[11] Hei Law and Jia Deng. Cornernet: Detecting objects as paired keypoints. 2019.

[12] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.

[13] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988, 2017.

[14] Yutian Lin, Liang Zheng, Zhedong Zheng, Yu Wu, Zhilan Hu, Chenggang Yan, and Yi Yang. Improving person re-identification by attribute and identity learning. *Pattern Recognition*, 2019.

[15] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In *European conference on computer vision*, pages 21–37. Springer, 2016.

[16] Davide Mazzini and Raimondo Schettini. Spatial sampling network for fast scene understanding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 0–0, 2019.

[17] Ishan Misra, C Lawrence Zitnick, and Martial Hebert. Shuffle and learn: unsupervised learning using temporal order verification. In *European Conference on Computer Vision*, pages 527–544. Springer, 2016.

[18] Alejandro Newell, Zhiao Huang, and Jia Deng. Associative embedding: End-to-end learning for joint detection and grouping. In *Advances in Neural Information Processing Systems*, pages 2277–2287, 2017.

[19] Mehdi Noroozi and Paolo Favaro. Unsupervised learning of visual representations by solving jigsaw puzzles. In *European Conference on Computer Vision*, pages 69–84. Springer, 2016.

[20] Deepak Pathak, Philipp Krahenbuhl, Jeff Donahue, Trevor Darrell, and Alexei A Efros. Context encoders: Feature learning by inpainting. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2536–2544, 2016.

[21] Sida Peng, Yuan Liu, Qixing Huang, Xiaowei Zhou, and Hujun Bao. Pvnet: Pixel-wise voting network for 6dof pose estimation. In *CVPR*, 2019.

[22] Joseph Redmon and Ali Farhadi. Yolo9000: better, faster, stronger. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7263–7271, 2017.

[23] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018.

[24] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016.

[25] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015.

[26] Pierre Sermanet, David Eigen, Xiang Zhang, Michaël Mathieu, Rob Fergus, and Yann LeCun. Overfeat: Integrated recognition, localization and detection using convolutional networks. *ICLR*, 2014.

[27] M. Teichmann, M. Weber, M. Zöllner, R. Cipolla, and R. Urtasun. Multinet: Real-time joint semantic reasoning for autonomous driving. In *2018 IEEE Intelligent Vehicles Symposium (IV)*, pages 1013–1020, June 2018. doi: 10.1109/IVS.2018.8500504.

[28] Fisher Yu, Dequan Wang, Evan Shelhamer, and Trevor Darrell. Deep layer aggregation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2403–2412, 2018.

[29] Richard Zhang, Phillip Isola, and Alexei A Efros. Colorful image colorization. In *European conference on computer vision*, pages 649–666. Springer, 2016.

[30] Shifeng Zhang, Longyin Wen, Xiao Bian, Zhen Lei, and Stan Z Li. Single-shot refinement neural network for object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4203–4212, 2018.

[31] Xingyi Zhou, Dequan Wang, and Philipp Krähenbühl. Objects as points. In *arXiv preprint arXiv:1904.07850*, 2019.

[32] Xingyi Zhou, Jiacheng Zhuo, and Philipp Krähenbühl. Bottom-up object detection by grouping extreme and center points. In *CVPR*, 2019.

Appendix A: Model Architectures

Figure 1 illustrates our model architecture with the ResNet-18 and ResNet-101 backbones. Figure 2 illustrates our model architecture with the DLA backbone.

Appendix B: Results on Pascal VOC

We compare the performance of our model with CenterNet on the Pascal VOC object detection dataset in Table 1.

Appendix C: Maximum IoU of a given box with arbitrary boxes centered at a given point

Let $B_1(0, 0, w_1, h_1)$ be a given (fixed) rectangular 2D box centered at $O(0, 0)$ with width w_1 and height h_1. Consider a variable box denoted by $B(\delta_x, \delta_y, w, h)$ that is centered at a fixed coordinate (δ_x, δ_y) but has variable height and width denoted by h and w, respectively. Figure 3 illustrates the boxes considered. Without any loss of generality, we assume $\delta_x \geq 0, \delta_y \geq 0$ throughout, unless otherwise explicitly specified. We consider a generic placement of two intersecting boxes, as shown in Figure 3, that satisfies

$$\delta_x - \frac{w}{2} > -\frac{w_1}{2}, \text{ and} \tag{1}$$

$$\delta_x - \frac{w}{2} < \frac{w_1}{2}, \text{ and} \tag{2}$$

$$\delta_x + \frac{w}{2} > \frac{w_1}{2}, \text{ and} \tag{3}$$

$$\delta_y - \frac{h}{2} > -\frac{h_1}{2}, \text{ and} \tag{4}$$

$$\delta_y - \frac{h}{2} < \frac{h_1}{2}, \text{ and} \tag{5}$$

$$\delta_y + \frac{h}{2} > \frac{h_1}{2}. \tag{6}$$

These simplify to the following succinct inequality conditions on w and h:

$$|w_1 - 2\delta_x| \ < \ w \ < \ w_1 + 2\delta_x, \tag{7}$$

$$|h_1 - 2\delta_y| \ < \ h \ < \ h_1 + 2\delta_y. \tag{8}$$

Unless otherwise specified, the above conditions are assumed to hold for the rest of the discussion.

Recall that the intersection-over-union (IoU) measure of two boxes B_1 and B is defined as

$$IoU(B_1, B) = I(B_1, B)/U(B_1, B),$$

where

$$I(B_1, B) = \text{Ar}(B_1 \cap B),$$

and

$$\begin{aligned} U(B_1, B) &= \text{Ar}(B_1 \cup B) \\ &= \text{Ar}(B_1) + \text{Ar}(B) - \text{Ar}(B_1 \cap B). \end{aligned}$$

Result In this section, we show that, given a pair of intersecting bounding boxes B_1 (fixed) and B (variable) in a generic placement (illustrated in Figure 3), the $IoU(B_1, B)$ as a function of w and h (dimensions of the variable box B) has no local minima or maxima. We show that the critical points (w^*, h^*) of $IoU(B_1, B)$ are, in fact, saddle points.

This, in turn, means that there cannot be a tuple $(\hat{w}, \hat{h})$ that maximizes $IoU(B_1, B)$ and simultaneously does not lie on the boundary of the space spanned by valid (w, h) tuples ("validity" is defined by the inequalities in Eq. (7) and (8), for instance). The points on the boundary for the generic placement of boxes chosen in Figure 3 can be written down explicitly as:

$$\begin{aligned} w &= |w_1 \pm 2\delta_x|, \text{ and} \\ h &= |h_1 \pm 2\delta_y|, \end{aligned}$$

which corresponds to four combinations of (w, h) tuples given by $\{(|w_1 + 2\delta_x|, \ |h_1 + 2\delta_y|), \ (|w_1 + 2\delta_x|, \ |h_1 - 2\delta_y|), \ (|w_1 - 2\delta_x|, \ |h_1 + 2\delta_y|), \ (|w_1 - 2\delta_x|, \ |h_1 - 2\delta_y|)\}$. Therefore, it is sufficient to check values at the boundaries when searching for $(\hat{w}, \hat{h})$ that maximizes $IoU(B_1, B)$. Notice that points on the boundary correspond to degenerate choices of B that have at least two of its edges collinear with the edges of the fixed box B_1.

Proof Consider the IoU of the boxes B_1 and B in Figure 3 as:

$$IoU = \frac{I}{\text{Ar}(B_1) + \text{Ar}(B) - I}.$$

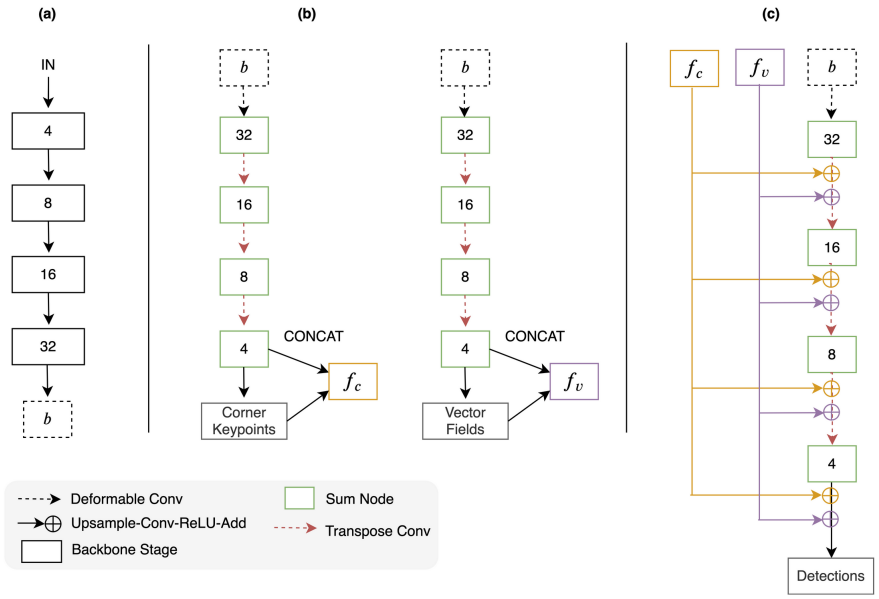


Figure 1: Our architecture with ResNet backbone. The numbers in the boxes represent the downsampling factors with respect to the input resolution. (a) ResNet backbone with output features denoted by b. (b) Decoder architecture for auxiliary predictions. Following CenterNet, deformable convolutions are used before each up-sampling layer. The backbone features b are shared among the two decoders for corner keypoint and vector field predictions. Concatenated features and predictions are denoted by f_c and f_v for each of the two auxiliary outputs. (c) Detector branch uses an identical decoder network except that the transformed features $\hat{T}(f_c)$ and $\hat{T}(f_b)$ are additively fused to each stage of transpose convolution, where $\hat{T}$ denotes upsampling followed by a 3×3 convolution and ReLU to keep the dimensions commensurate for addition. Note that the CenterNet architecture, in comparison, consists only of the ResNet backbone (shown in (a)) and a decoder (shown in (c)) without any additive fusion operations.

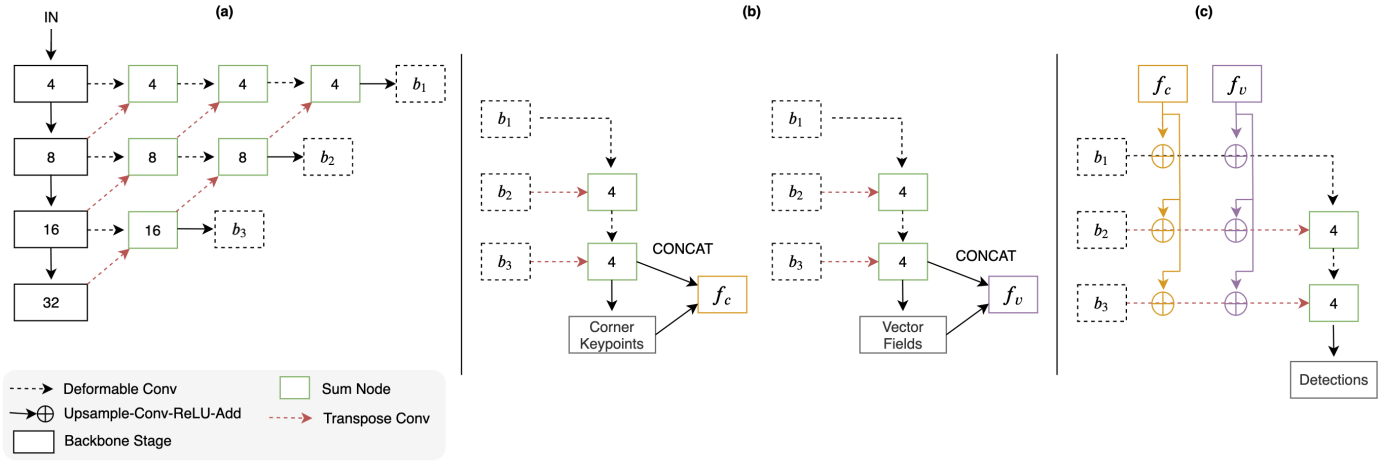


Figure 2: Our architecture with DLA backbone. The numbers in the boxes represent the downsampling factors with respect to the input resolution. (a) DLA backbone with output features at multiple resolutions denoted by $\{b_1, b_2, b_3, b_4\}$. Following CenterNet, we use deformable convolutions in the upsampling stages. (b) *Iterative deep aggregation* (IDA) blocks are used as decoders. The backbone features $\{b_i\}$ are shared among the decoders. Concatenated features and predictions are denoted by f_c and f_v for the corner keypoints and vector fields, respectively. (c) Detector branch uses an identical IDA block for estimating the center heatmap and the size offsets. The backbone features $\{b_i\}$ are additively fused with the transformed features $\hat{T}_i(f_c)$ and $\hat{T}_i(f_b)$, where $\hat{T}_i$ denotes upsampling to match b_i's resolution followed by a 3×3 convolution and a ReLU nonlinearity. CenterNet architecture, in comparison, consists only of the DLA backbone (shown in (a)) and an IDA decoder (shown in (c)) without any additive fusion operations.

Backbone	Init	Model	mAP_{384}	mAP_{512}	FPS_{384}	FPS_{512}
DLA34	ImageNet	CenterNet	77.92 / 79.23	78.90 / 80.31	66.6 / 43.5	47.6 / 27.7
		Ours	**78.63 / 79.85**	**80.24 / 81.46**	47.6 / 30.3	35.7 / 19.2
ResNet101	ImageNet	CenterNet	75.85 / 77.10	77.46 / 78.96	50.0 / 34.5	34.5 / 23.3
		Ours	**76.14 / 77.67**	**78.09 / 79.84**	40.0 / 27.7	28.6 / 18.5
ResNet18	ImageNet	CenterNet	70.22 / 72.41	71.82 / 74.65	125 / 100	111 / 77.0
		Ours	**71.16 / 73.45**	**74.13 / 76.53**	91.0 / 66.7	71.4 / 45.5

Table 1: Evaluation of networks on Pascal VOC *test2007*. We report mAP (at an IoU threshold of 0.5) without test-time augmentations as well as with horizontal flip averaging. The input resolution is kept fixed at 384×384 or 512×512, following CenterNet. Timings are reported on our system with a NVIDIA P100 GPU.

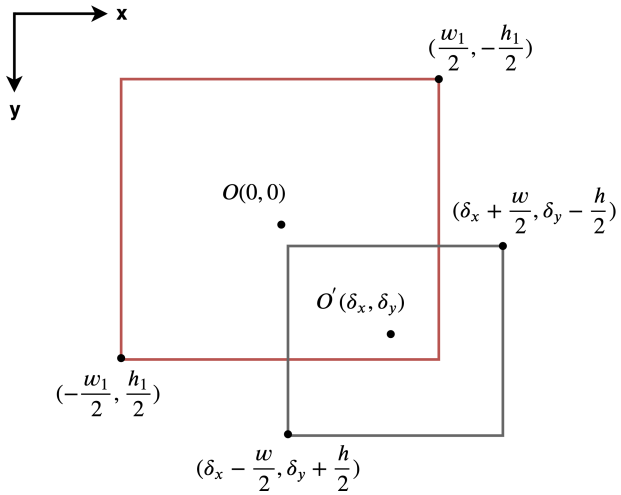


Figure 3: Generic placement of a pair of bounding boxes that intersect. The given (fixed) box is denoted by $B_1(0, 0, w_1, h_1)$ centered at $O(0, 0)$ (drawn in red). A variable box (with variable height h and width w) is centered at a given point (δ_x, δ_y) and is denoted by $B(\delta_x, \delta_y, w, h)$ (drawn in grey). Without any loss of generality, δ_x, δ_y are assumed to be greater than or equal to zero. Centers and two diagonally opposite corners (bottom-left and top-right corners) of the boxes are labeled.

This can be rewritten as

$$IoU = \frac{\frac{I}{\text{Ar}(B_1)+\text{Ar}(B)}}{1 - \frac{I}{\text{Ar}(B_1)+\text{Ar}(B)}} = \frac{S}{1-S},$$

where

$$S \equiv \frac{I}{\text{Ar}(B_1) + \text{Ar}(B)}.$$

Since $f(x) = \frac{x}{1-x}$ is a strictly increasing function in the interval $x \in [0, 1)$, one can conclude that $IoU = \frac{S}{1-S}$ is strictly increasing in S since $0 \leq S < 1$. Therefore, IoU and S share the same set of stationary and inflexion points. We work with S instead of IoU due to its simpler form.

For the case illustrated in Figure 3, one may write the area of intersection I as

$$I = \left[\frac{w_1}{2} - \left(\delta_x - \frac{w}{2}\right)\right]\left[\frac{h_1}{2} - \left(\delta_y - \frac{h}{2}\right)\right].$$

Therefore, S can be written as:

$$S = \frac{\left[\frac{w_1}{2} - \left(\delta_x - \frac{w}{2}\right)\right]\left[\frac{h_1}{2} - \left(\delta_y - \frac{h}{2}\right)\right]}{w_1 h_1 + wh}.$$

Computing the gradient of S with respect to w and h, we

have the following:

$$\frac{\partial S}{\partial (w,h)} = \frac{1}{4(w_1 h_1 + wh)^2} \Bigg((h_1 + h - 2\delta_y)(w_1 h_1 - w_1 h + 2h\delta_x), \\ (w_1 + w - 2\delta_x)(w_1 h_1 - h_1 w + 2w\delta_y) \Bigg). \tag{9}$$

Therefore, for stationary points we have:

$$\frac{\partial S}{\partial w} = 0 \implies w_1 h_1 - w_1 h + 2h\delta_x = 0,$$
$$\frac{\partial S}{\partial h} = 0 \implies w_1 h_1 - h_1 w + 2w\delta_y = 0,$$

since $h_1 + h - 2\delta_y > 0$ (from Eq. (5)) and $w_1 + w - 2\delta_x > 0$ (from Eq. (2)).

It is easy to see that the symmetric second-order derivatives, namely, $\frac{\partial^2 S}{\partial w^2}$ and $\frac{\partial^2 S}{\partial h^2}$, vanish at the above stationary points. Specifically, one can easily show that $\frac{\partial^2 S}{\partial w^2} = 0$ whenever $\frac{\partial S}{\partial w} = 0$. Similarly, $\frac{\partial^2 S}{\partial h^2} = 0$ whenever $\frac{\partial S}{\partial h} = 0$.

But the asymmetric second-order derivative $\frac{\partial^2 S}{\partial h \partial w}$ does not vanish for generic cases. Therefore, the determinant of the Hessian of S is negative at the stationary points, implying that these are saddle points.

Hence, we conclude that the optimal height h and width w for B that maximizes its IoU with B_1 must lie on the boundary since the stationary points are saddle points.

Appendix D: Qualitative Examples

Qualitative results of our auxiliary predictions (corner heatmaps and vector-fields) and final detections are shown in Figures 4, 5, and 6 for a few samples in MS COCO test-dev dataset.

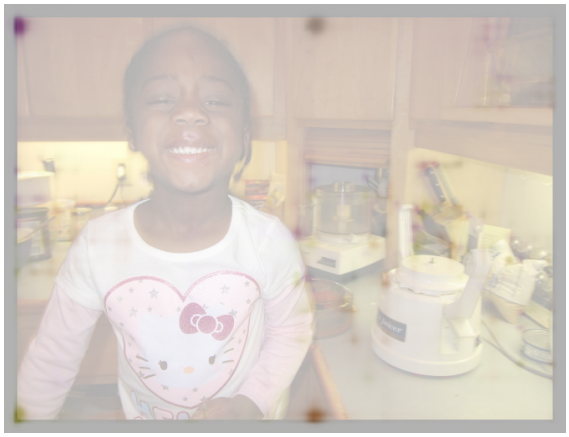


Figure 4: Illustration of the auxiliary heatmap predictions of the corner keypoints. Image 296903 from MS COCO test-dev, run on our best DLA34-2x model. The top-left, top-right, bottom-left and bottom-right corners are represented by purple, grey, greenish-yellow and orange colors, respectively.

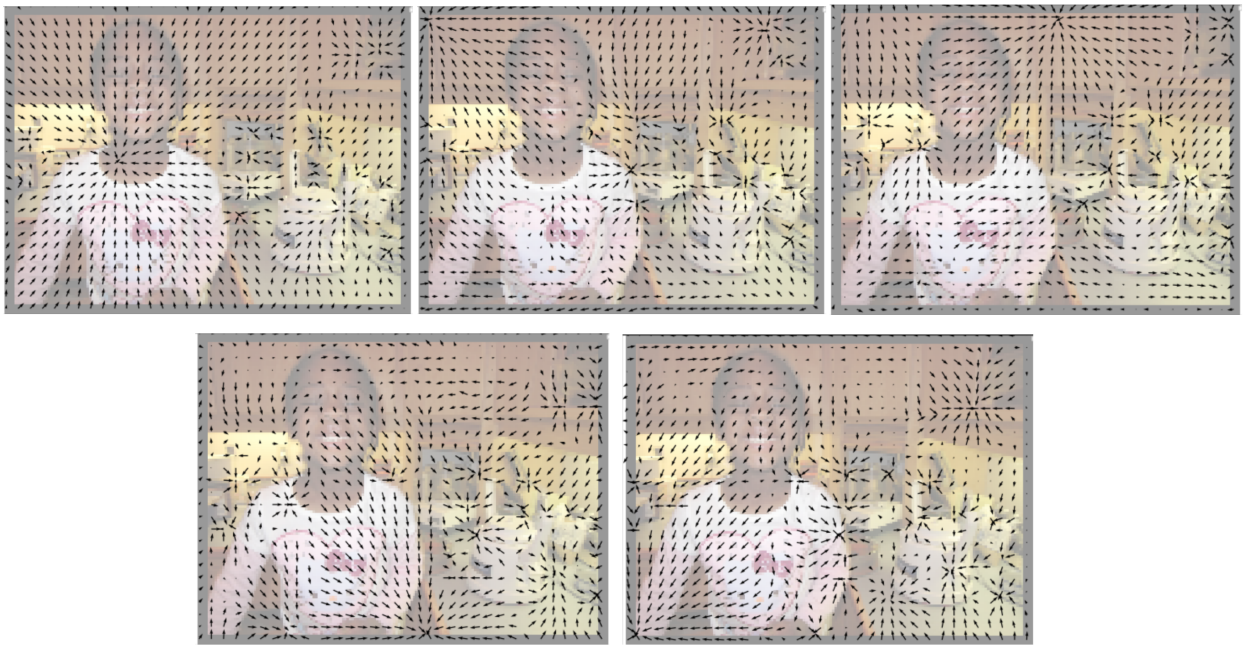


Figure 5: We show a few examples of the predicted vector-field outputs on a MS COCO test-dev image 296903 using our best DLA34-2x model. The top row illustrates the predicted direction-fields for the center, top-left, and top-right corners, respectively. The bottom row shows direction-fields voting for bottom-right and bottom-left corner keypoints.

Figure 6: We show a few qualitative examples of detections on MS COCO test-dev images using our best DLA34-2x model. Detections with center keypoint confidence scores < 0.3 are suppressed from the visualization.