Object as point summary

Abstract

Detection identifies objects as axis-aligned boxes in an image. Most successful object detectors enumerate a nearly exhaustive list of potential object locations and classify each. This is wasteful, inefficient, and requires additional post-processing. In this paper, we take a different approach. We model an object as a single point — the center point of its bounding box. Our detector uses keypoint estimation to find center points and regresses to all other object properties, such as size, 3D location, orientation, and even pose. Our center point based approach, CenterNet, is end-to-end differentiable, simpler, faster, and more accurate than corresponding bounding box based detectors. CenterNet achieves the best speed-accuracy trade-off on the MS COCO dataset, with 28.1% AP at 142 FPS, 37.4% AP at 52 FPS, and 45.1% AP with multi-scale testing at 1.4 FPS. We use the same approach to estimate 3D bounding box in the KITTI benchmark and human pose on the COCO keypoint dataset. Our method performs competitively with sophisticated multi-stage methods and runs in real-time.

▲検出は、画像内の軸に沿ったボックスとしてオブジェクトを識別します。成功しているほとんどのオブジェクト検出器は、ほぼ網羅的なオブジェクト位置のリストを列挙し、それぞれを分類しています。これは無駄が多く、非効率的であり、追加の後処理を必要とします。この論文では、別のアプローチを取ります。我々は、物体を1つの点、つまり外接箱の中心点としてモデル化します。我々の検出器は、キーポイント推定を使用して中心点を見つけ、サイズ、3D位置、向き、ポーズなど、他のすべてのオブジェクトの特性に回帰します。当社の中心点ベースのアプローチであるCenterNetは、対応するバウンディングボックスベースの検出器よりも、エンドツーエンドで微分可能で、シンプル、高速、高精度です。CenterNetは、MS COCOのデータセットにおいて、142 FPSで28.1%のAP、52 FPSで37.4%のAP、1.4 FPSでのマルチスケールテストで45.1%のAPと、最高の速度と精度のトレードオフを達成しています。KITTIベンチマークの3Dバウンディングボックスと COCOキーポイントデータセットの人間のポーズを推定するために同じアプローチを使用しています。この手法は、高度な多段階手法と競合し、リアルタイムでの実行が可能である。

Introduction

Object detection powers many vision tasks like instance segmentation [7,21,32], pose estimation [3, 15, 39], tracking [24, 27], and action recognition [5]. It has down-stream applications in surveillance [57], autonomous driving [53], and visual question answering [1]. Current object detectors represent each object through an axis-aligned bounding box that tightly encompasses the object [18,19,33, 43, 46]. They then reduce object detection to image classification of an extensive number of potential object bounding boxes. For each bounding box, the classifier determines if the image content is a specific object or background. Onestage detectors [33, 43] slide a complex arrangement of possible bounding boxes, called anchors, over the image and classify them directly without specifying the box content. Two-stage detectors [18, 19, 46] recompute image features for each potential box, then classify those features. Post-processing, namely non-maxima suppression, then removes duplicated detections for the same instance by computing bounding box IoU. This post-processing is hard to differentiate and train [23], hence most current detectors are not end-to-end trainable. Nonetheless, over the past five years [19], this idea has achieved good empirical success [12,21,25,26,31,35,47,48,56,62,63]. Sliding window based object detectors are however a bit wasteful, as they need to enumerate all possible object locations and dimensions.

▲物体検出は，インスタンスのセグメンテーション[7,21,32]，ポーズ推定[3,15,39]，トラッキング[24,27]，行動認識[5]など，多くのビジョンタスクをサポートしています． また，監視 [57]，自律運転 [53]，視覚的な質問応答 [1] などの分野でも応用されています． 現在の物体検出器は，各物体を軸に沿ったバウンディングボックスで表現し，物体を厳密に包んでいます[18,19,33,43,46]． そして，物体検出を，膨大な数の潜在的な物体バウンディングボックスの画像分類にまで落とし込みます．各バウンディングボックスについて，分類器は画像の内容が特定の物体か背景かを判断します．ワンステージ検出器 [33, 43] は，アンカーと呼ばれる複雑な配置のバウンディングボックスを画像上にスライドさせ，ボックスの内容を指定せずに直接分類します．2段階検出器 [18, 19, 46] は，各可能性のあるボックスについて画像特徴を再計算し，それらの特徴を分類する． その後，後処理，すなわち非最大化抑制処理が行われ，バウンディングボックスIoUを計算することで，同じインスタンスについて重複した検出を除去する．[12,21,25,26,31,35,47,48,56,62,63]．この後処理は，区別して訓練するのが難しいため[23]，現在のほとんどの検出器は，エンドツーエンドで訓練することができません． それにもかかわらず，過去5年間[19]で，このアイデアは経験的に良い成功を収めている しかし，スライディングウィンドウベースの物体検出器は，可能なすべての物体の位置と寸法を列挙する必要があるため，少し無駄が多い．

In this paper, we provide a much simpler and more efficient alternative. We represent objects by a single point at their bounding box center [see Figure 2]. Other properties, such as object size, dimension, 3D extent, orientation, and pose are then regressed directly from image features at the center location. Object detection is then a standard keypoint estimation problem [3,39,60]. We simply feed the input image to a fully convolutional network [37,40] that generates a heatmap. Peaks in this heatmap correspond to object centers. Image features at each peak predict the objects bounding box height and weight. The model trains using standard dense supervised learning [39,60]. Inference is a single network forward-pass, without non-maximal suppression for post-processing.

▲この論文では，よりシンプルで効率的な代替案を提供します．我々は，外接箱の中心にある一点でオブジェクトを表現します[図2を参照]．次に，物体のサイズ，寸法，3次元の広がり，向き，ポーズなどの他の特性を，中心位置の画像特徴から直接回帰させます．物体検出は，標準的なキーポイント推定問題[3,39,60]になります．我々は単に入力画像を完全畳み込みネットワーク[37,40]に送るだけで，ヒートマップを生成します．このヒートマップのピークは物体の中心に対応しています．各ピークの画像特徴は，物体の境界ボックスの高さと重さを予測します．モデルは，標準的な密な教師付き学習[39,60]を使用して学習します．推論は単一のネットワークフォワードパスであり，後処理のための非最大抑制はありません．

Our method is general and can be extended to other tasks with minor effort. We provide experiments on 3D object detection [17] and multi-person human pose estimation [4], by predicting additional outputs at each center point [see Figure 4]. For 3D bounding box estimation, we regress to the object absolute depth, 3D bounding box dimensions, and object orientation [38]. For human pose estimation, we consider the 2D joint locations as offsets from the center and directly regress to them at the center point location. The simplicity of our method, Center Net, allows it to run at a very high speed [Figure 1]. With a simple Resnet18 and up-convolutional layers [55], our network runs at 142 FPS with 28.1% COCO bounding box AP. With a carefully designed keypoint detection network, DLA34 [58], our network achieves 37.4% COCO AP at 52 FPS. Equipped with the state-of-the-art keypoint estimation network, Hourglass-104 [30, 40], and multi-scale testing, our network achieves 45.1% COCO AP at 1.4 FPS. On 3D bounding box estimation and human pose estimation, we perform competitively with state-of-the-art at a higher inference speed. Code is available at [here] [https://github.com/xingyizhou/CenterNet.].

▲我々の手法は一般的であり，少しの労力で他のタスクにも拡張可能である．各中心点での追加出力を予測することで，3 次元物体検出 [17] と複数人の人間の姿勢推定 [4] の実験を行う[図 4 参照]．3 次元バウンディングボックス推定では，物体の絶対深度，3 次元バウンディングボックスの寸法，物体の向き[38]に回帰する．人間の姿勢推定では，2 次元の関節位置を中心からのオフセットとみなし，中心点の位置で直接回帰します． 我々の手法であるCenter Netのシンプルさにより、非常に高速に実行することができます[図1]。シンプルなResnet18とアップコンバージョン層[55]を使用すると、私たちのネットワークは、28.1%のCOCOCOバウンディングボックスAPを使用して142 FPSで動作します。慎重に設計されたキーポイント検出ネットワークDLA34[58]では、我々のネットワークは52 FPSで37.4%のCOCOCO APを達成しています。最新鋭のキーポイント推定ネットワークHourglass-104[30, 40]を搭載し、マルチスケールテストを実施した結果、1.4FPSで45.1%のCOCO APを達成した。また、3次元バウンディングボックス推定や人物の姿勢推定では、より高い推論速度で最先端技術と同等の性能を実現しています。コードはここで公開されています。

▶ 参考サイト

Object as Points slide

[コンピュータビジョンの最新論文調査 キーポイントによる物体検出編]
[https://engineer.dena.com/posts/2019.07/cv-papers-19-keypoint-object-detection/]

[最近のSingle Shot系の物体検出のアーキテクチャまとめ][https://www.slideshare.net/ren4yu/single-shot]

[物体検出についての歴史まとめ][https://qiita.com/mshinoda88/items/9770ee671ea27f2c81a9]

[Recent Advances in Deep Learning for Object Detection - Part 1][https://www.dlology.com/blog/recent-advances-in-deep-learning-for-object-detection/]

[Non-Maximum Suppressionを世界一わかりやすく解説する][https://meideru.com/archives/3538]

[エンドツーエンド深層学習のフロンティア][https://app.journal.ieice.org/trial/101_9/k101_9_920/index.html]

Related work

Object detection by region classification.

One of the first successful deep object detectors, RCNN [19], enumerates object location from a large set of region candidates [52], crops them, and classifies each using a deep network. Fast-RCNN [18] crops image features instead, to save computation. However, both methods rely on slow low-level region proposal methods.

▲最初に成功した深層物体検出器の1つであるRCNN[19]は、大規模な領域候補のセット[52]から物体の位置を列挙し、それらを切り出し、深層ネットワークを使用してそれぞれを分類します。高速RCNN [18]は、計算量を節約するために、代わりに画像特徴を切り出します。しかし，どちらの手法も低速な低レベル領域提案法に依存している．

Object detection with implicit anchors.

Faster RCNN [46] generates region proposal within the detection network. It samples fixed-shape bounding boxes [anchors] around a low-resolution image grid and classifies each into "foreground or not". An anchor is labeled foreground with a >0.7 overlap with any ground truth object, background with a <0.3 overlap, or ignored otherwise. Each generated region proposal is again classified [18].

▲高速RCNN [46] は，検出ネットワーク内で領域提案を生成する．この手法は，低解像度の画像グリッドの周囲にある固定形状の境界ボックス[アンカー]をサンプリングし，それぞれを「前景か非前景か」に分類する．アンカーは、0.7以上の重なりがある場合は前景、0.3未満の重なりがある場合は背景、それ以外の場合は無視されるとラベル付けされる。生成された各領域提案は、再び分類される[18]。

Changing the proposal classifier to a multi-class classification forms the basis of one-stage detectors. Several improvements to one-stage detectors include anchor shape priors [44, 45], different feature resolution [36], and loss re-weighting among different samples [33]. Our approach is closely related to anchor-based onestage approaches [33, 36, 43]. A center point can be seen as a single shape-agnostic anchor [see Figure 3]. However, there are a few important differences. First, our CenterNet assigns the "anchor" based solely on location, not box overlap [18]. We have no manual thresholds [18] for foreground and background classification. Second, we only have one positive "anchor" per object, and hence do not need NonMaximum Suppression [NMS] [2]. We simply extract local peaks in the keypoint heatmap [4,39]. Third, CenterNet uses a larger output resolution [output stride of 4] compared to traditional object detectors [21, 22] [output stride of 16]. This eliminates the need for multiple anchors [47].

▲提案分類器をマルチクラス分類に変更することは、1段検出器の基礎を形成する。1段検出器へのいくつかの改良点は、アンカー形状プリオール[44, 45]、異なる特徴分解能[36]、および異なるサンプル間の損失再重み付け[33]を含む。 我々のアプローチは、アンカーベースのワンステージアプローチ[33, 36, 43]と密接に関連している。中心点は、単一の形状に依存しないアンカーとして見ることができる[図３参照]。しかし，いくつかの重要な違いがある．第一に，我々の CenterNet は，箱の重なりではなく，位置のみに基づいて「アンカー」を割り当てる [18]．我々は、前景と背景の分類のための手動のしきい値[18]を持っていません。第二に、我々は1つのオブジェクトに対して1つの正の「アンカー」しか持たないため、NonMaximum Suppression [NMS] [2]を必要としない。我々は，単にキーポイントヒートマップの局所的なピークを抽出するだけである[4,39]．第三に、CenterNetは、従来のオブジェクト検出器[21, 22][出力ストライド16]と比較して、より大きな出力分解能[出力ストライド4]を使用しています。これにより，複数のアンカーが不要となる[47].

Object detection by keypoint estimation.

We are not the first to use keypoint estimation for object detection. CornerNet [30] detects two bounding box corners as keypoints, while ExtremeNet [61] detects the top-, left-, bottom-, rightmost, and center points of all objects. Both these methods build on the same robust keypoint estimation network as our Center Net. However, they require a combinatorial grouping stage after keypoint detection, which significantly slows down each algorithm. Our CenterNet, on the other hand, simply extracts a single center point per object without the need for grouping or post-processing.

▲キーポイント推定による物体検出

物体検出にキーポイント推定を使用したのは我々が初めてではない．また，ExtremeNet [61] は，すべての物体の上点，左点，下点，右端，中央点を検出している．これらの手法はいずれも，我々の Center Net と同じロバストなキーポイント推定ネットワークをベースにしている．しかし，これらの手法では，キーポイント検出後にコンビナトリアルグルーピングの段階を必要とするため，各アルゴリズムの処理速度が大幅に低下する．一方、我々のCenterNetは、グループ化や後処理を必要とせず、オブジェクトごとに単一の中心点を抽出するだけです。

Monocular 3D object detection.

3D bounding box estimation powers autonomous driving [17]. Deep3Dbox [38] uses a slow-RCNN [19] style framework, by first detecting 2D objects [46] and then feeding each object into a 3D estimation network. 3D RCNN [29] adds an additional head to Faster-RCNN [46] followed by a 3D projection. Deep Manta [6] uses a coarse-to-fine Faster-RCNN [46] trained on many tasks. Our method is similar to a one-stage version of Deep3Dbox [38] or 3DRCNN [29]. As such, CenterNet is much simpler and faster than competing methods.

▲単眼3次元物体検出

3Dバウンディングボックス推定は自律走行に威力を発揮します[17]．Deep3Dbox [38]は、最初に2Dオブジェクト[46]を検出してから、各オブジェクトを3D推定ネットワークに送り込むことで、低速RCNN [19]スタイルのフレームワークを使用しています。3D RCNN [29]は、Faster-RCNN [46]に追加のヘッドを追加し、その後3D投影を行う。Deep Manta [6]は、多くのタスクで訓練された粗から細までのFaster-RCNN [46]を使用しています。我々の手法は、Deep3Dbox[38]や3DRCNN[29]の1段階バージョンに似ています。このように、CenterNetは競合する手法よりもはるかにシンプルで高速である。

Preliminary

Let $I \in R^{W \times H \times 3}$ be an input image of width W and height $H .$ Our aim is to produce a keypoint heatmap $\hat{Y} \in$ $[0,1]*{R}^{\underline{w}} \times \frac{H}{R} \times C,$ where R is the output stride and C is the number of keypoint types. Keypoint types include $C=17$ human joints in human pose estimation $[4,55],$ or $C=80$ object categories in object detection $[30,61] .$ We use the default output stride of $R=4$ in literature $[4,40,42] .$ The out put stride downsamples the output prediction by a factor R A prediction $\hat{Y}*{x, y, c}=1$ corresponds to a detected keypoint while $\hat{Y}_{x, y, c}=0$ is background. We use several different fully-convolutional encoder-decoder networks to predict $\hat{Y}$ from an image $I:$ A stacked hourglass network $[30,40],$ upconvolutional residual networks (ResNet) [22,55], and deep layer aggregation (DLA) [58].

▲予備的な

IRWH3を幅W、高さHの入力画像とする。 ここで，R は出力ストライド，C はキーポイントタイプの数である．キーポイントタイプには，人間のポーズ推定(4,55)ではC = 17個の人間の関節，物体検出ではC = 80個の物体カテゴリが含まれます[30,61]．我々は、文献(4,40,42)にあるR=4のデフォルトの出力ストライドを使用する。出力ストライドは，出力予測を係数Rでダウンサンプリングする．予測値Y=1は検出されたキーポイントに対応し、Y=0はバックグラウンドである。画像Ｉから Ｙを予測するために、いくつかの異なる完全畳み込みエンコーダ／デコーダネットワークを使用する：積層砂時計ネットワーク（３０，４０）、アップコンボリューション残差ネットワーク（ＲｅｓＮｅｔ）［２２，５５］、および深層凝集（ＤＬＡ）［５８］。

We train the keypoint prediction network following Law and Deng [30]. For each ground truth keypoint $p \in \mathcal{R}^{2}$ of class $c,$ we compute a low-resolution equivalent $\tilde{p}=\left\lfloor\frac{p}{R}\right\rfloor$. We then splat all ground truth keypoints onto a heatmap $Y \in[0,1]*{R}^{\underline{W}} \times \frac{H}{R} \times C$ using a Gaussian ker nel $Y*{x y c}=\exp \left(-\frac{\left(x-\tilde{p}*{x}\right)^{2}+\left(y-\tilde{p}*{y}\right)^{2}}{2 \sigma_{p}^{2}}\right),$ where σ_{p} is an object size-adaptive standard deviation $[30] .$ If two Gaussians of the same class overlap, we take the element-wise maximum [4]. The training objective is a penalty-reduced pixelwise logistic regression with focal loss [33]

$$L_{k}=\frac{-1}{N} \sum_{x y c}\left{\begin{array}{cc} \left(1-\hat{Y}{x y c}\right)^{\alpha} \log \left(\hat{Y}{x y c}\right) & \text { if } Y_{x y c}=1 \ \left(1-Y_{x y c}\right)^{\beta}\left(\hat{Y}{x y c}\right)^{\alpha} & \text { otherwise } \ \log \left(1-\hat{Y}{x y c}\right) \end{array}\right.$$

▲Law and Deng (30)に従ってキーポイント予測ネットワークを学習する。クラスcの各基底真理キーポイントPR2について，低解像度の等価PPRを計算する．次に、ガウスカーネル Y = exp() を用いて、すべての基底真理キーポイントをヒートマップ Y 上にスプラットします（シグマはオブジェクトサイズ適応標準偏差[30]）。同じクラスの2つのガウシアンが重なり合う場合は、要素ごとの最大値を取る [4]。学習目的は，焦点損失を伴うペナルティを軽減したピクセル単位のロジスティック回帰である[33].

where α and β are hyper-parameters of the focal loss [33] and N is the number of keypoints in image I. The normalization by N is chosen as to normalize all positive focal loss instances to $1 .$ We use $\alpha=2$ and $\beta=4$ in all our experiments, following Law and Deng [30].

To recover the discretization error caused by the output stride, we additionally predict a local offset $\hat{O} \in \mathcal{R} \frac{W}{R} \times \frac{H}{R} \times 2$ for each center point. All classes c share the same offset prediction. The offset is trained with an L1 loss $$ L_{o f f}=\frac{1}{N} \sum_{p}\left|\hat{O}_{\tilde{p}}-\left(\frac{p}{R}-\tilde{p}\right)\right|$$

src/lib/models/losses.py

```
class RegL1Loss(nn.Module):
  def __init__(self):
    super(RegL1Loss, self).__init__()

  def forward(self, output, mask, ind, target):
    pred = _transpose_and_gather_feat(output, ind)
    mask = mask.unsqueeze(2).expand_as(pred).float()
    # loss = F.l1_loss(pred * mask, target * mask, reduction='elementwise_mean')
    loss = F.l1_loss(pred * mask, target * mask, size_average=False)
    loss = loss / (mask.sum() + 1e-4)
    return loss
```

ctdet.py

```
    self.crit_reg = RegL1Loss() if opt.reg_loss == 'l1' else \
              RegLoss() if opt.reg_loss == 'sl1' else None
```

```
      if opt.reg_offset and opt.off_weight > 0:
        off_loss += self.crit_reg(output['reg'], batch['reg_mask'],
                                  batch['ind'], batch['reg']) / opt.num_stacks
```

The supervision acts only at keypoints locations $\tilde{p}$, all other locations are ignored.

In the next section, we will show how to extend this keypoint estimator to a general purpose object detector.

▲ここで、 a とＢは焦点損失のハイパーパラメータ（３３）であり、 Ｎは画像Ｉのキーポイントの数である。 Nによる正規化は、すべての正の焦点損失インスタンスを1に正規化するように選択される。我々は、LawとDeng（30）に倣って、全ての実験においてa = 2とb = 4を使用する。

出力ストライドに起因する離散化誤差を回復するために、各中心点について局所的なオフセットORWH2を追加で予測する。すべてのクラスcは同じオフセット予測を共有している。オフセットは、L1 損失です.

この監視は、キーポイントの位置pでのみ作用し、それ以外の位置は無視される。

次節では、このキーポイント推定器を汎用的な物体検出器に拡張する方法を示す。

▶ 参考サイト

[[論文紹介] Focal Loss for Dense Object Detection][https://qiita.com/agatan/items/53fe8d21f2147b0ac982]

Objects as Points

Let $\left(x_{1}^{(k)}, y_{1}^{(k)}, x_{2}^{(k)}, y_{2}^{(k)}\right)$ be the bounding box of ob ject k with category $c_{k} .$ Its center point is lies at $p_{k}=$ $\left(\frac{x_{1}^{(k)}+x_{2}^{(k)}}{2}, \frac{y_{1}^{(k)}+y_{2}^{(k)}}{2}\right) .$ We use our keypoint estimator $\hat{Y}$ to predict all center points. In addition, we regress to the object size $s_{k}=\left(x_{2}^{(k)}-x_{1}^{(k)}, y_{2}^{(k)}-y_{1}^{(k)}\right)$ for each object k To limit the computational burden, we use a single size prediction $\hat{S} \in \mathcal{R}^{\frac{W}{R} \times \frac{H}{R} \times 2}$ for all object categories. We use an L1 loss at the center point similar to Objective 2 : $$ L_{s i z e}=\frac{1}{N} \sum_{k=1}^{N}\left|\hat{S}_{p_{k}}-s_{k}\right| $$ We do not normalize the scale and directly use the raw pixel coordinates. We instead scale the loss by a constant $\lambda_{s i z e}$ The overall training objective is $$ L_{d e t}=L_{k}+\lambda_{s i z e} L_{s i z e}+\lambda_{o f f} L_{o f f} $$

ctdet.py

```
    loss = opt.hm_weight * hm_loss + opt.wh_weight * wh_loss + \
           opt.off_weight * off_loss
```

We set $\lambda_{s i z e}=0.1$ and $\lambda_{o f f}=1$ in all our experiments unless specified otherwise. We use a single network to predict the keypoints $\hat{Y}$, offset $\hat{O},$ and size $\hat{S}$. The network predicts a total of $C+4$ outputs at each location. All outputs share a common fully-convolutional backbone network. For each modality, the features of the backbone are then passed through a separate 3×3 convolution, ReLU and another 1×1 convolution. Figure 4 shows an overview of the network output. Section 5 and supplementary material contain additional architectural details.

▲(x1, y1, x2, y2)をカテゴリCkの物体kの外接箱とする。その中心点は Pk にある。すべての中心点を予測するために、我々のキーポイント推定器yを使用する。さらに、各物体 k について、物体サイズ sk = (x2-x1, y2-y1) に回帰します。計算負荷を制限するために、すべての物体カテゴリについて単一のサイズ予測 SEWH2 を使用します。中心点には、目的２と同様のＬ１損失を用いる。

我々はスケールを正規化せず、生のピクセル座標を直接使用します。その代わりに、損失を一定のサイズでスケーリングします。全体的な学習目的は LDET です.

我々は，特に指定がない限り，すべての実験においてλ=0.1とλ=1を設定した．キーポイントY、オフセットO、サイズSを予測するために単一のネットワークを使用します。 このネットワークは、各場所で合計C+4個の出力を予測します。 すべての出力は、共通の完全畳み込みバックボーンネットワークを共有する。各モダリティについて、バックボーンの特徴は、次に、別個の3×3畳み込み、ReLU、および別の1×1畳み込みに渡される。図４は、ネットワーク出力の概要を示す。セクション５および補足資料には、追加のアーキテクチャの詳細が記載されています。

From points to bounding boxes

At inference time, we first extract the peaks in the heatmap for each category independently. We detect all responses whose value is greater or equal to its 8-connected neighbors and keep the top 100 peaks. Let PC be the set of n detected center points P of class C. Each keypoint location is given by an integer coordinates (xi, yi). We use the keypoint values Y me as a measure of its detection confidence, and produce a bounding box at location

▲点からバウンディングボックスへ.

推論時には、まず、各カテゴリのヒートマップのピークを独立して抽出します。値が8個の隣人と同じかそれ以上の値を持つすべての応答を検出し、上位100個のピークを保持します。PCをクラスCのn個の検出された中心点Pの集合とする。各キーポイントの位置は整数の座標(xi, yi)で与えられる。検出信頼度の指標としてキーポイントの値Y meを使用し、その位置にバウンディングボックスを生成します。