DiffusionDet: Diffusion Model for Object Detection

Zixuan Pan

10/10/2023

Outline

1. Introduction to object detection

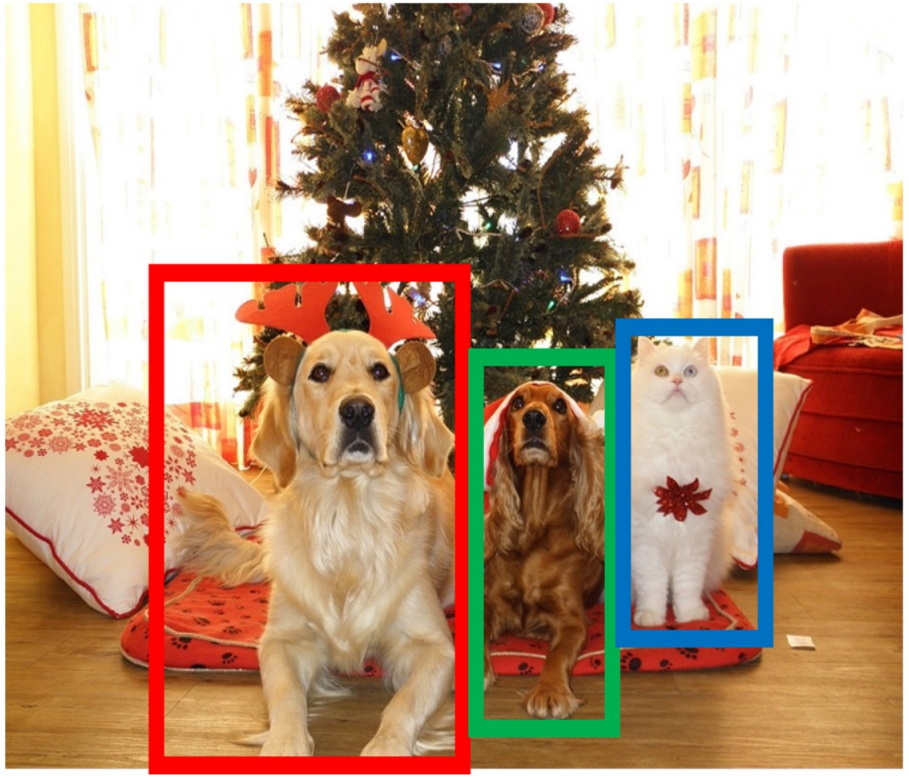
2. DiffusionDet paper

Object Detection: Overview

Input: single RGB image

Output: A set of detected objects

Each one has a bounding box (x, y, h, w), and category label



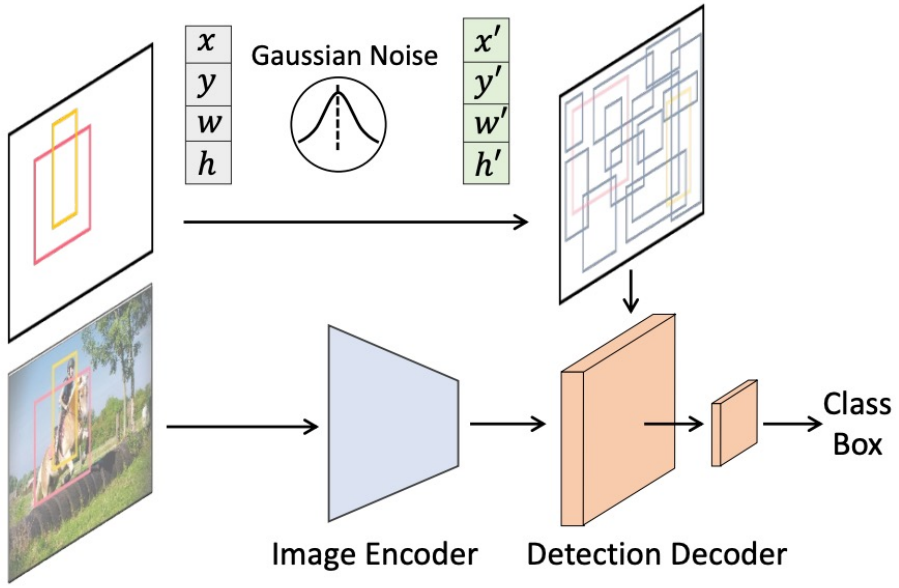
Object Detection: Training Scheme

High level: Backbone network

Development of object candidates

Two prediction heads

(Class + Box)



Chen, Shoufa, et al. "Diffusiondet: Diffusion model for object detection."
Proceedings of the IEEE/CVF International Conference on Computer Vision. 2023.

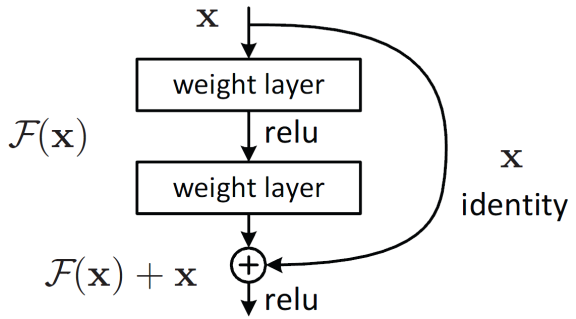
Backbone Networks

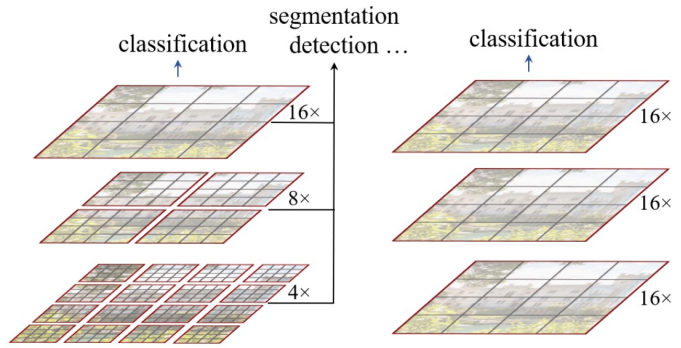
ResNet

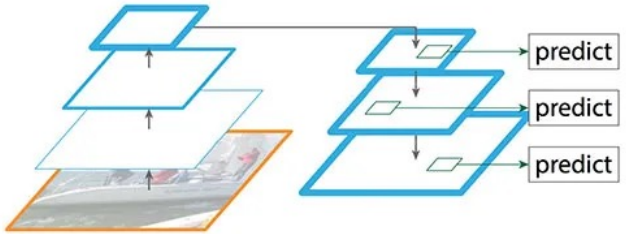
Swin Transformer

Feature Pyramid Network (FPN)

Multiscale networks are preferred.







[1] He, Kaiming, et al. "Deep residual learning for image recognition." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016.

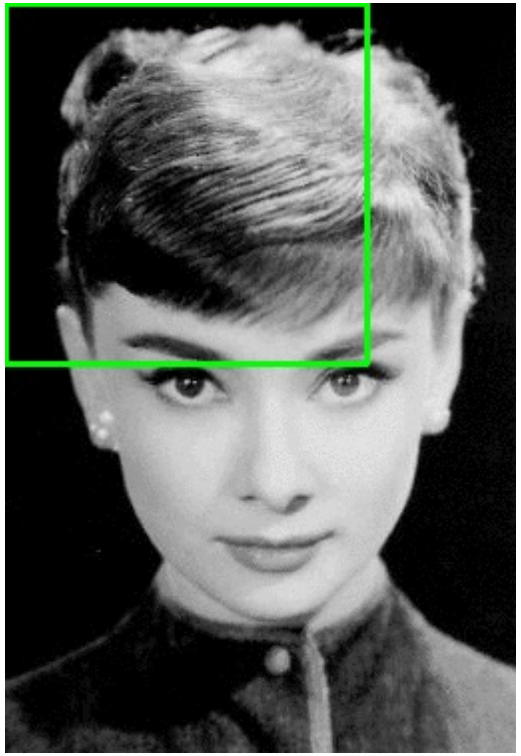
[2] Liu, Ze, et al. "Swin transformer: Hierarchical vision transformer using shifted windows." *Proceedings of the IEEE/CVF international conference on computer vision*. 2021.

[3] Lin, Tsung-Yi, et al. "Feature pyramid networks for object detection." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017.

Development of object candidates

Sliding Window

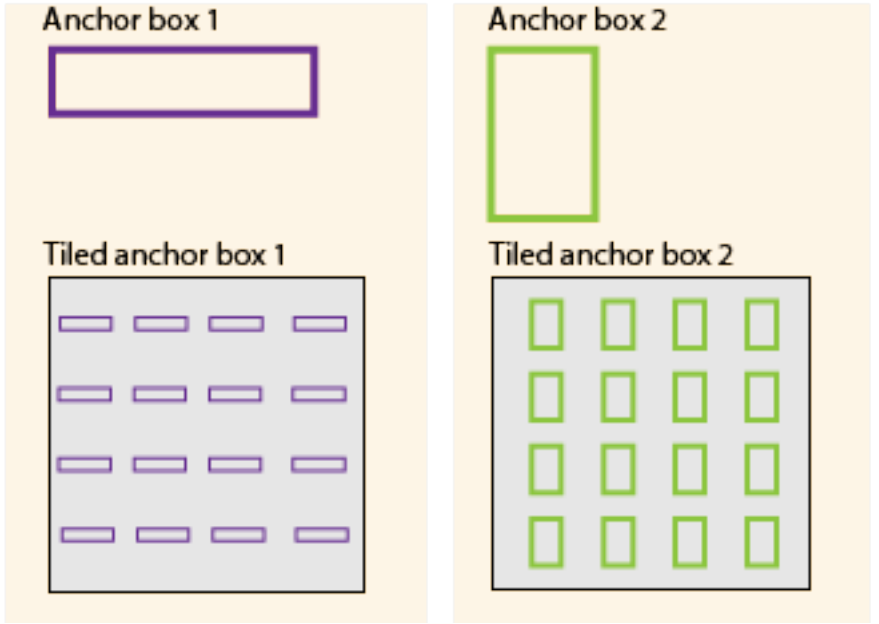
- Slide over the image with a fixed box size.
- The computation is very heavy.



Development of object candidates

Anchor Box

- A set of boxes with pre-defined width and height.
- To determine which boxes to choose, a common method is to get important “anchors” first using cluster methods.



Development of object candidates

Region proposal

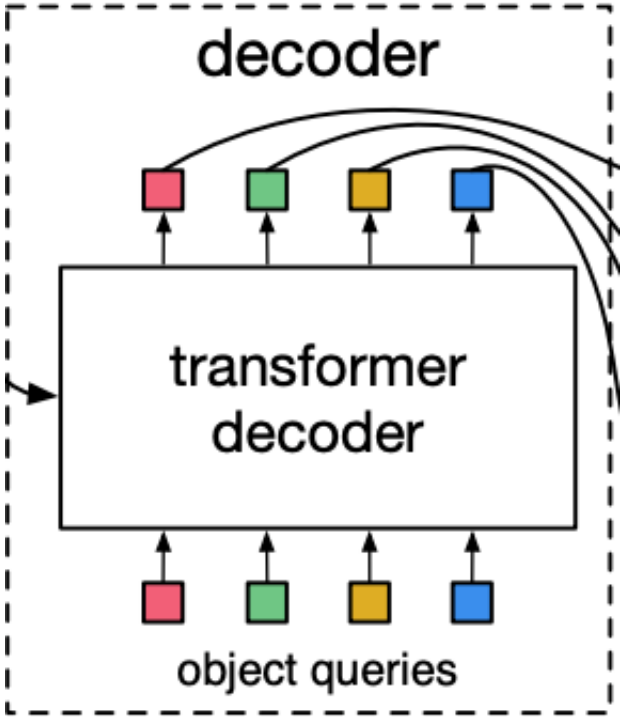
- Find a small set of boxes that are likely to cover all objects (still larger than actually needed, ~2000 in general).
- Based on heuristics, for example “blob-like” images/ supervised learning to classify whether an anchor is in the object or not.
- Much faster to compute.

Girshick, Ross, et al. "Rich feature hierarchies for accurate object detection and semantic segmentation." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2014.

Development of object candidates

Learnable Embedding

- Pass a large number of learnable tokens into Transformer. Concatenate each token with image embeddings.
- Run Hungarian matching to find the corresponding GT boxes.
- No need for extra prior knowledge or classifier. Change the prediction from known priors to arbitrary inputs.
- Suffer from domain gap, would fail if the number of bounding boxes in training set and test set varies.

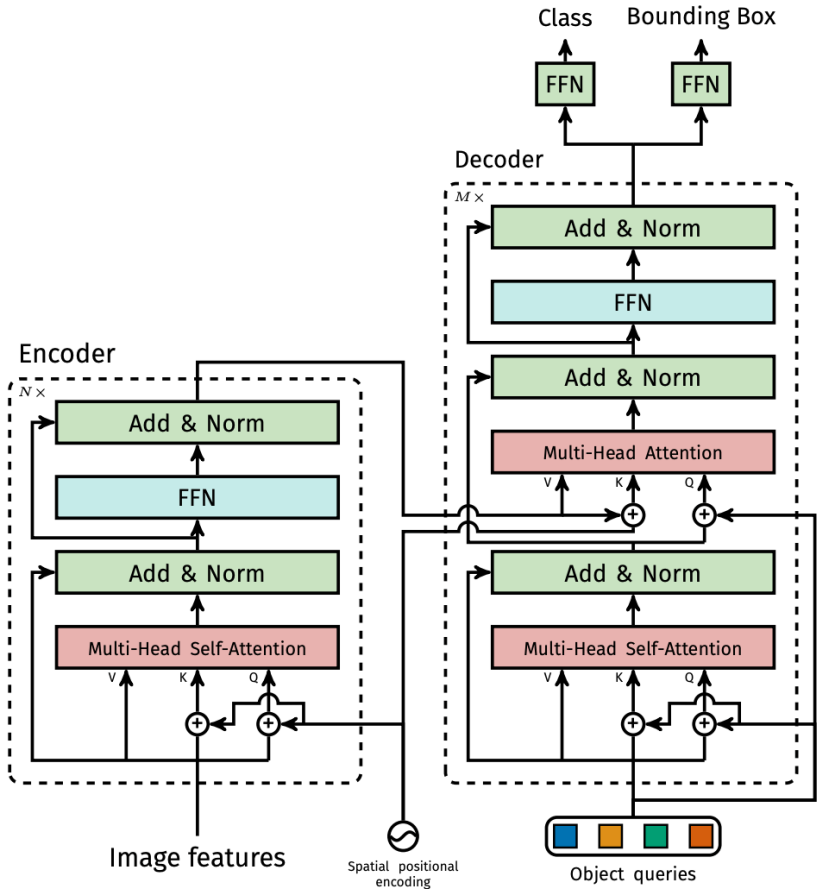


Carion, Nicolas, et al. "End-to-end object detection with transformers."
European conference on computer vision. Cham: Springer International Publishing, 2020.

Development of object candidates

Learnable Embedding

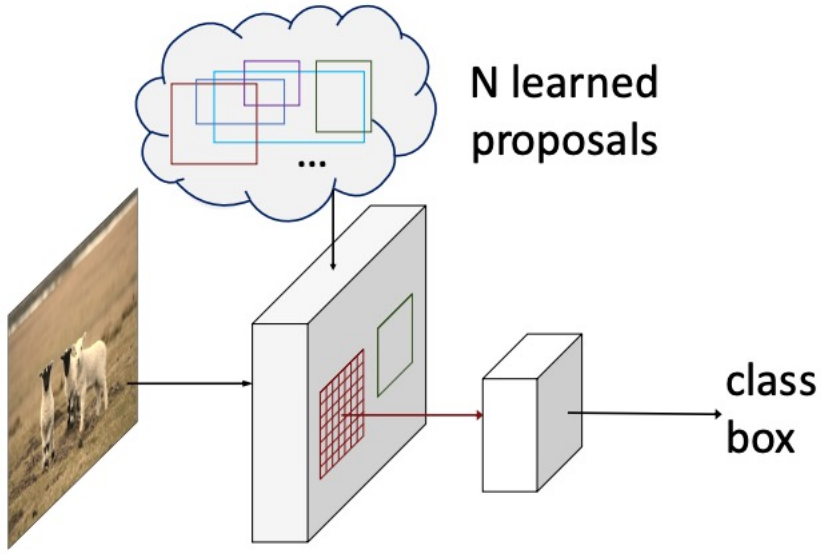
- Pass a large number of learnable tokens into Transformer. Concatenate each token with image embeddings.
- Run Hungarian matching to find the corresponding GT boxes.
- No need for extra prior knowledge or classifier. Change the prediction from known priors to arbitrary inputs.



Development of object candidates

Learnable Embedding

- An alternative way is to use learned proposals instead of randomly initialized embeddings as inputs to the decoder.
- The most significant difference is that it use box coordinates instead of arbitrary size embeddings.
- The learned proposals can be seen as statistics of box coordinates in the training set.



Sun, Peize, et al. "Sparse r-cnn: End-to-end object detection with learnable proposals."
Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2021.

Prediction Heads

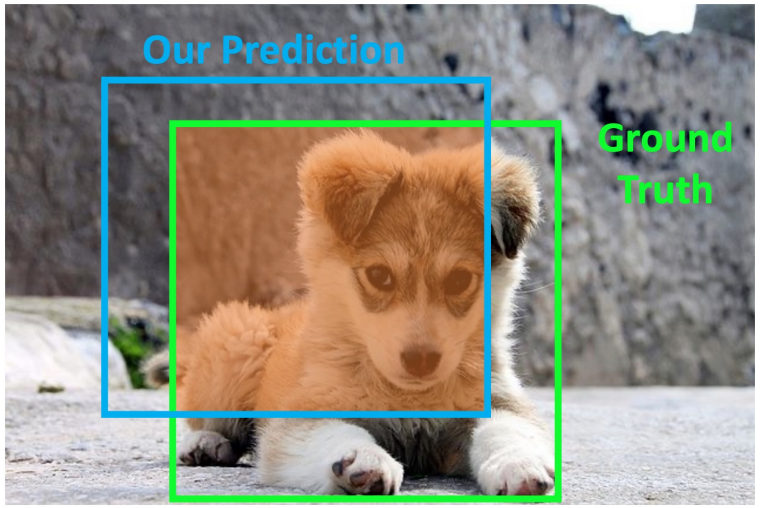
Two separate FFNs to predict class label and box coordinates respectively

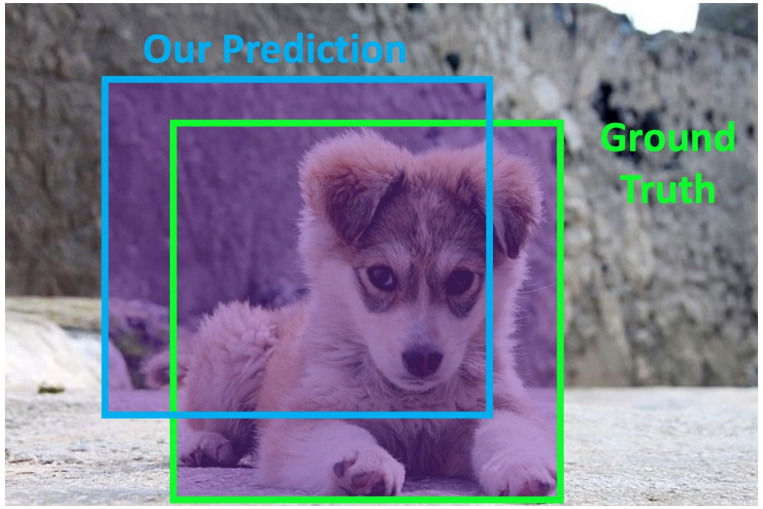
- Class prediction is optimized with Cross Entropy loss.
- Sometimes need to add a “background” class.
- Box prediction is optimized with L2/smooth L1 loss to predict (x, y, w, h).

Object Detection: Evaluation

Intersection of union (IOU)

IOU = Area of intersection / Area of Union





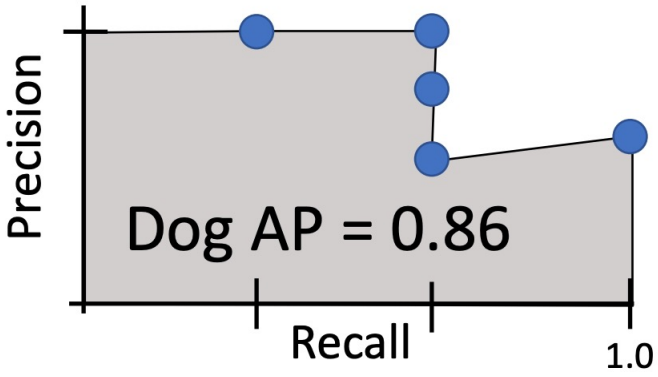
Object Detection: Evaluation

Mean Average Precision(mAP) is the most widely used metric.

For each label category, starting from most likely prediction:

- Judge whether IOU of prediction box and ground truth box is larger than a threshold.
- If so, mark as positive, other wise negative.
- Mark the point on a precision-recall graph.
- AP for this class is the area under precision recall curve

Compute the mean over all classes.



Challenges in Object Detection

- Images are in higher resolution than classification tasks
- A large redundancy in object candidates proposal, could make it slow to compute.
- The objects are in multiscale.

DiffusionDet: Overview

- Use image conditioned diffusion model to generate bounding box priors.

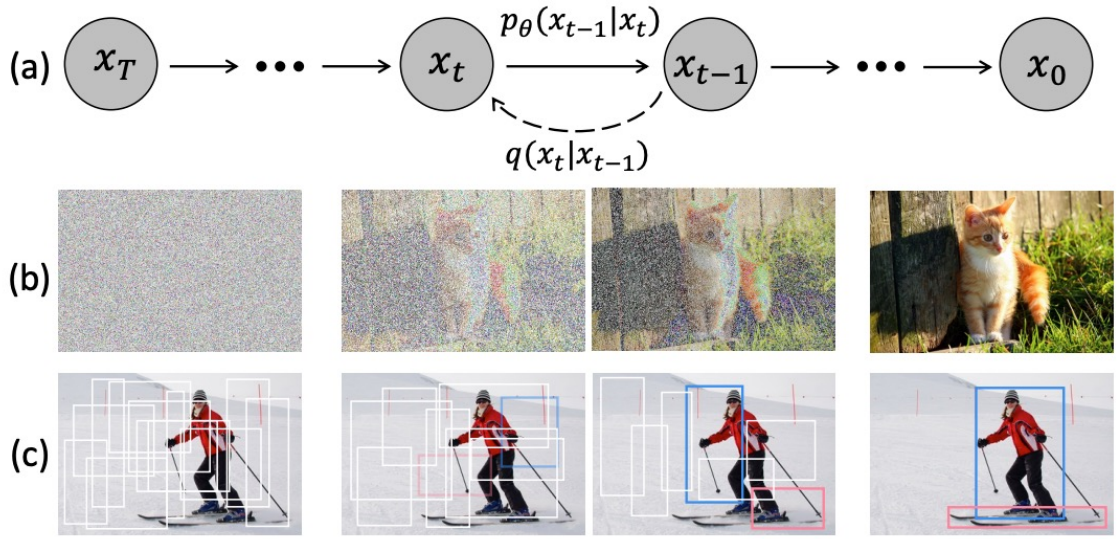
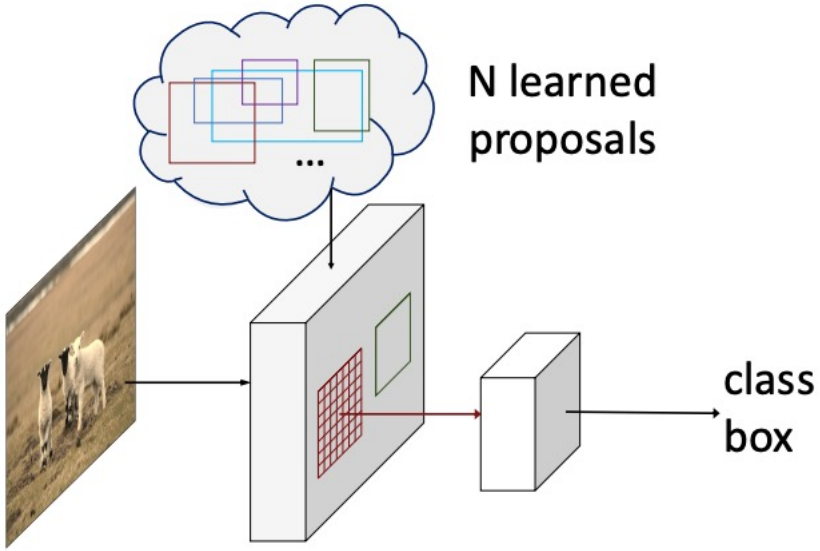
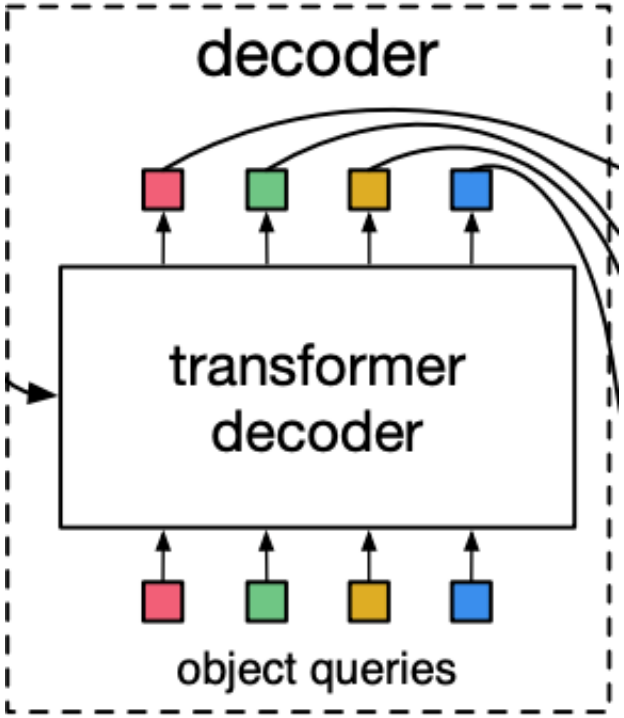


Figure 1. **Diffusion model for object detection**. (a) A diffusion model where q is the diffusion process and p_θ is the reverse process. (b) Diffusion model for image generation task. (c) We propose to formulate object detection as a denoising diffusion process from noisy boxes to object boxes.

DiffusionDet: Motivation

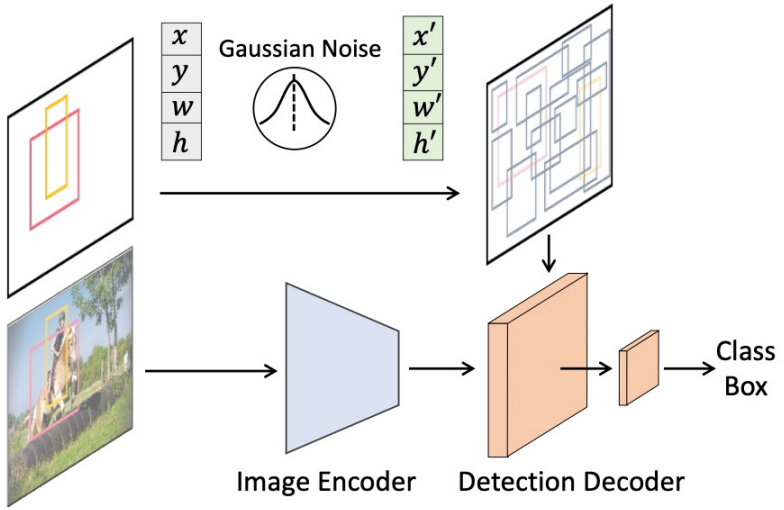
- Learnable embedding methods use a set of learnable embeddings of fixed size, make it much less flexible during inference. A small number of boxes during inference could takes less resource, and a large number might benefit the model.





DiffusionDet: Motivation

- DiffusionDet proposes using to generate non-learnable box priors sending to the decoder.



$$q(\boldsymbol{z}_t|\boldsymbol{z}_0) = \mathcal{N}(\boldsymbol{z}_t|\sqrt{\bar{\alpha}_t}\boldsymbol{z}_0, (1-\bar{\alpha}_t)\boldsymbol{I}),$$

$$\mathcal{L}_{\text{train}} = \frac{1}{2}||f_\theta(\boldsymbol{z}_t, t) - \boldsymbol{z}_0||^2.$$

DiffusionDet: Tricks

Ground Truth Box Padding

- Adding some noisy boxes aside the ground truth boxes to a fixed size N so that it can be trained on batch. Different strategies have been explored, including repeating existing ground truth boxes, concatenating random boxes or image-size boxes.

Box renewal

- In each sampling step, a score is calculated to represent how far each generated box is to real boxes. If the score is too low, they replace the box with a random one.

DiffusionDet: Algorithm

Algorithm 1 DiffusionDet Training

```
def train_loss(images, gt_boxes):
  """
  images: [B, H, W, 3]
  gt_boxes: [B, *, 4]
  # B: batch
  # N: number of proposal boxes
  """

  # Encode image features
  feats = image_encoder(images)

  # Pad gt_boxes to N
  pb = pad_boxes(gt_boxes) # padded boxes: [B, N, 4]

  # Signal scaling
  pb = (pb * 2 - 1) * scale

  # Corrupt gt_boxes
  t = randint(0, T)              # time step
  eps = normal(mean=0, std=1) # noise: [B, N, 4]
  pb_crpt = sqrt(    alpha_cumprod(t)) * pb +
            sqrt(1 - alpha_cumprod(t)) * eps

  # Predict
  pb_pred = detection_decoder(pb_crpt, feats, t)

  # Set prediction loss
  loss = set_prediction_loss(pb_pred, gt_boxes)

  return loss
```

`alpha_cumprod(t)`: cumulative product of α_i, *i.e.*, $\prod_{i=1}^{t} \alpha_i$

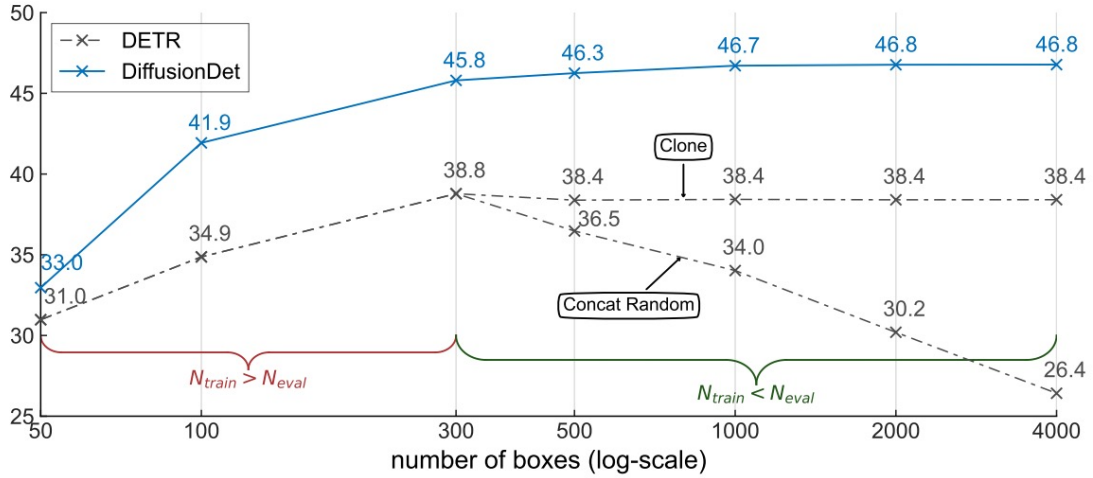
Algorithm 2 DiffusionDet Sampling

```
def infer(images, steps, T):
  """
  images: [B, H, W, 3]
  # steps: number of sample steps
  # T: number of time steps
  """

  # Encode image features
  feats = image_encoder(images)

  # noisy boxes: [B, N, 4]
  pb_t = normal(mean=0, std=1)

  # uniform sample step size
  times = reversed(linespace(-1, T, steps))

  # [(T-1, T-2), (T-2, T-3), ..., (1, 0), (0, -1)]
  time_pairs = list(zip(times[:-1], times[1:])

  for t_now, t_next in zip(time_pairs):
    # Predict pb_0 from pb_t
    pb_pred = detection_decoder(pb_t, feats, t_now)

    # Estimate pb_t at t_next
    pb_t = ddim_step(pb_t, pb_pred, t_now, t_next)

    # Box renewal
    pb_t = box_renewal(pb_t)

  return pb_pred
```

`linespace`: generate evenly spaced values

DiffusionDet: Experiments

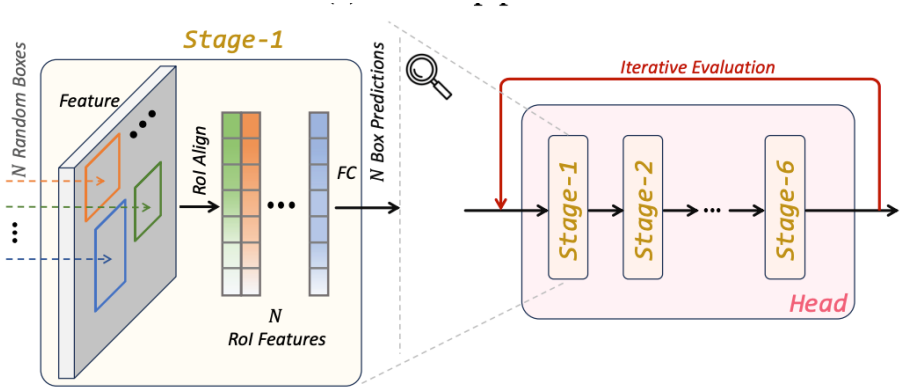
Comparing with DETR on 300 training boxes and different numbers of evaluation boxes. To modify the DETR evaluation box numbers, both clone and random concatenating method are used.



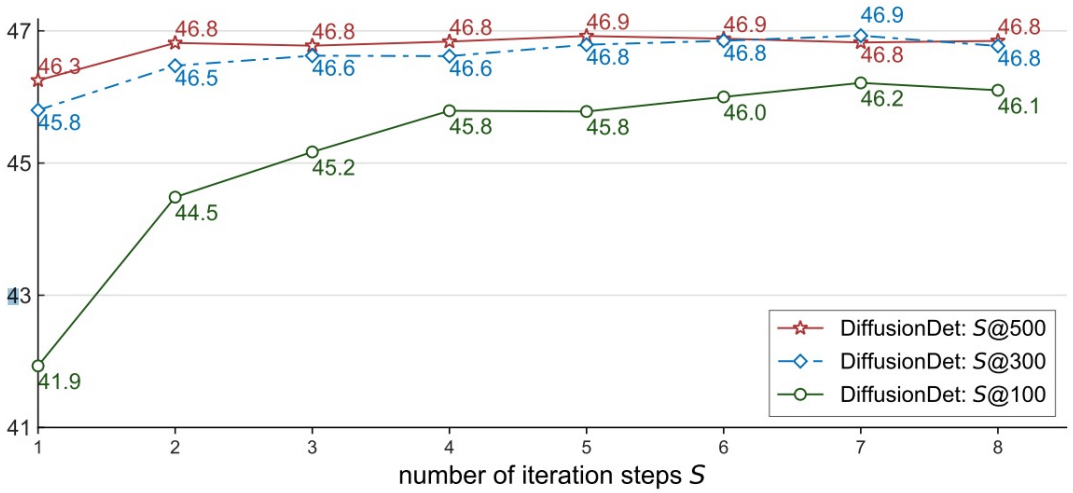
(a) **Dynamic number boxes.** Both DETR and DiffusionDet are trained with 300 object queries or proposal boxes. More proposal boxes in inference bring accuracy improvement on DiffusionDet, while degenerate DETR.

DiffusionDet: Experiments

Inspired by Cascade RCNN, the authors tested iterative evaluation strategy, which sends the predicted box into decoder for multiple times. They found that it helps much with a fewer random boxes.



(b) Details of the detection decoder/head.



(b) **Iterative evaluation.** 'S@500' denotes that we evaluate DiffusionDet with 500 boxes using a different number of iteration steps. For all cases, the accuracy increases with refinement times.

DiffusionDet: Experiments

Full benchmark with other methods on different datasets and backbones

Method	AP	AP_{50}	AP_{75}	AP_s	AP_m	AP_l
			ResNet-50 [37]			
RetinaNet [102]	38.7	58.0	41.5	23.3	42.3	50.3
Faster R-CNN [102]	40.2	61.0	43.8	24.2	43.5	52.0
Cascade R-CNN [102]	44.3	62.2	48.0	26.6	47.7	57.7
DETR [10]	42.0	62.4	44.2	20.5	45.8	61.1
Deformable DETR [115]	43.8	62.6	47.7	26.4	47.1	58.0
Sparse R-CNN [91]	45.0	63.4	48.2	26.9	47.2	59.5
DiffusionDet (1 @ 300)	45.8	64.1	50.4	27.6	48.7	62.2
DiffusionDet (4 @ 300)	46.6	65.1	51.3	28.9	49.2	62.1
DiffusionDet (1 @ 500)	46.3	64.8	50.7	28.6	49.0	62.1
DiffusionDet (4 @ 500)	46.8	65.3	51.8	29.6	49.3	62.2
			ResNet-101 [37]			
RetinaNet [102]	40.4	60.2	43.2	24.0	44.3	52.2
Faster R-CNN [102]	42.0	62.5	45.9	25.2	45.6	54.6
Cascade R-CNN [11]	45.5	63.7	49.9	27.6	49.2	59.1
DETR [10]	43.5	63.8	46.4	21.9	48.0	61.8
Sparse R-CNN [91]	46.4	64.6	49.5	28.3	48.3	61.6
DiffusionDet (1 @ 300)	46.7	65.0	51.0	29.6	49.7	63.2
DiffusionDet (4 @ 300)	47.4	65.8	52.0	30.1	50.4	63.1
DiffusionDet (1 @ 500)	47.2	65.7	51.6	30.2	50.2	62.7
DiffusionDet (4 @ 500)	47.5	65.7	52.0	30.8	50.4	63.1
			Swin-Base [60]			
Cascade R-CNN [60]	51.9	70.9	56.5	35.4	55.2	67.4
Sparse R-CNN	52.0	72.2	57.0	35.8	55.1	68.2
DiffusionDet (1 @ 300)	52.5	71.8	57.3	35.0	56.4	69.3
DiffusionDet (4 @ 300)	53.3	72.8	58.6	36.6	57.0	69.2
DiffusionDet (1 @ 500)	53.0	72.3	58.0	35.5	56.9	69.1
DiffusionDet (4 @ 500)	53.3	72.7	58.4	36.2	56.9	69.0

Table 2. **Comparisons with different object detectors on COCO 2017 `val` set**. `[S@`N_{eval}`]` denotes the number of iteration steps `S` and number of evaluation boxes N_{eval}. The reference after each method indicates the source of its results. The method without reference is our implementation.

Method	AP	AP_{50}	AP_{75}	AP_r	AP_c	AP_f
			ResNet-50 [37]			
Faster R-CNN†	22.5	37.1	23.6	9.9	21.1	29.7
Cascade R-CNN†	26.3	37.8	27.8	12.3	24.9	34.1
Faster R-CNN	25.2	40.6	26.9	16.4	23.4	31.1
Cascade R-CNN	29.4	41.4	30.9	20.0	27.7	35.4
Sparse R-CNN	29.2	41.0	30.7	20.6	27.7	34.6
DiffusionDet (1 @ 300)	29.4	40.4	31.0	22.7	27.2	34.7
DiffusionDet (1 @ 500)	30.5	42.1	32.1	23.3	28.1	36.3
DiffusionDet (1 @ 1000)	31.4	43.2	33.3	24.5	28.8	37.3
DiffusionDet (4 @ 300)	31.5	43.4	33.5	24.1	29.3	37.4
			ResNet-101 [37]			
Faster R-CNN†	24.8	39.8	26.1	13.7	23.1	31.5
Cascade R-CNN†	28.6	40.1	30.1	15.3	27.3	35.9
Faster R-CNN	27.2	42.9	29.1	18.8	25.4	33.0
Cascade R-CNN	31.6	43.8	33.4	23.9	29.8	37.0
Sparse R-CNN	30.1	42.0	31.9	23.5	27.5	35.9
DiffusionDet (1 @ 300)	30.9	42.1	32.6	22.4	29.9	35.8
DiffusionDet (1 @ 500)	31.8	43.7	33.6	23.5	30.2	37.3
DiffusionDet (1 @ 1000)	33.0	45.0	34.9	24.8	31.4	38.3
DiffusionDet (4 @ 300)	33.0	45.2	35.1	24.2	31.5	38.5
			Swin-Base [60]			
DiffusionDet (1 @ 300)	39.5	52.3	42.0	33.0	38.5	43.5
DiffusionDet (1 @ 500)	40.8	54.2	43.6	33.4	39.9	45.2
DiffusionDet (1 @ 1000)	41.9	55.7	44.8	35.3	40.6	46.2
DiffusionDet (4 @ 300)	42.0	55.8	44.9	34.8	40.9	46.4

Table 3. **Comparisons with different object detectors on LVIS v1.0 `val` set**. We re-implement all detectors using federated loss [112] except for the rows in light gray (with †).

DiffusionDet: Experiments

Ablation studies on diffusion signal scale, GT box padding and sampling strategy.

scale	AP	AP_{50}	AP_{75}
0.1	38.9	54.3	42.1
1.0	45.0	63.0	48.9
2.0	45.8	64.1	50.4
3.0	45.6	63.9	50.0

(a) **Signal scale**. A large scaling factor can improve detection performance.

case	AP	AP_{50}	AP_{75}
Repeat	44.2	62.0	48.3
Cat Gaussian	45.8	64.1	50.4
Cat Uniform	45.2	63.3	49.3
Cat Full	45.7	63.9	49.9

(b) **GT boxes padding.** Concatenating Gaussian boxes works best.

DDIM	box renewal	iter 1	iter 2	iter 3
		45.8	44.4	44.1
✓		45.8	46.0	46.1
	✓	45.8	46.3	46.3
✓	✓	45.8	46.5	46.6

(c) **Sampling strategy.** Using both DDIM and box renewal works best.

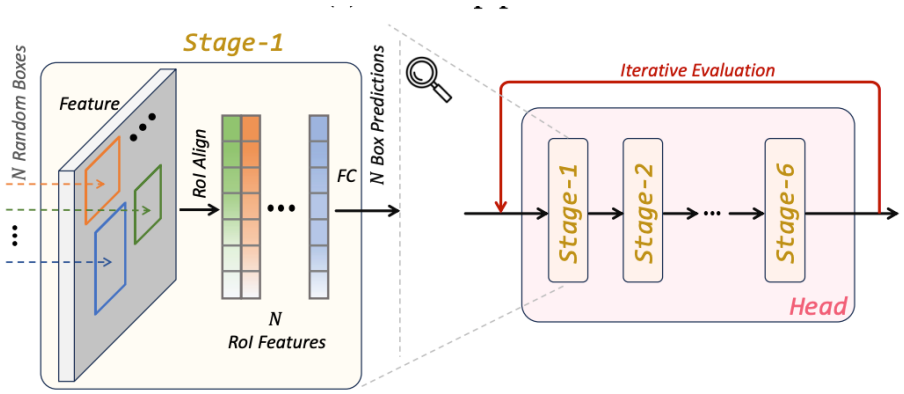
Table 4. **DiffusionDet ablation experiments** on COCO. We report AP, AP_{50}, and AP_{75}. If not specified, the default setting is: the backbone is ResNet-50 [37] with FPN [55], the signal scale is 2.0, ground-truth boxes padding method is concatenating Gaussian random boxes, DDIM and box renewal are used in sampling step. Default settings are marked in gray.

DiffusionDet: Experiments

Time-performance trade off compared with DETR and Sparse RCNN during training.

Method	Train	Test	COCO	CrowdHuman	FPS
DETR [10]	6×1	6×1	42.0	61.3	39
	6×1	6×2	41.6 (-0.4)	62.5 (+1.2)	32
Sparse R-CNN [91]	6×1	6×1	45.0	66.6	30
	6×1	6×2	43.6 (-1.4)	60.6 (-6.0)	21
	12×1	12×1	44.7 (-0.3)	66.1 (-0.5)	21
DiffusionDet	6×1	6×1	45.8	66.6	30
	6×1	6×2	46.5 (+0.7)	69.7 (+3.1)	20
DiffusionDet †	6×1	6×1	46.8 (+1.0)	71.0 (+4.4)	24

Table 6. **Running time *vs*. performance.** † denotes DiffusionDet with 1000 boxes. `#Stages×#Heads` denotes the number of `stages` and `heads` utilized during training and test phases. The definitions of `Stage` and `Head` are illustrated in Figure 2b.



(b) Details of the detection decoder/head.

DiffusionDet: Summary

Contributions

- The authors propose a new method that eliminates the requirement that the number of proposals must match for training and evaluation. They also achieved overall better results on several datasets.
- The methods seems benefits more from scaling up.

Limitations

- The sampling speed of diffusion model is too slow, make it unable to be applied on real-time scenarios.
- The smallest model with one iteration and 300 boxes doesn’t have a significant improvement towards previous methods.

Quiz time

1. Which part of object detection scheme is diffusion used for?

Quiz time

1. Which part of object detection scheme is diffusion used for?

Development of object candidates

Quiz time

2. In the process of developing object candidates, why can we start from feature maps for deep learning methods, given that the feature map and raw image are in different shape? (Hint: Think of the size of feature and the characteristic of backbone networks, you can take CNN as an example.)

Quiz time

2. In the process of developing object candidates, why can we start from feature maps for deep learning methods, given that the feature map and raw image are in different shape?

The height and width are of the same ratio. Meanwhile, the convolution/local attention will only be calculated with nearby regions. You can think of each point in feature map corresponds to an anchor box.