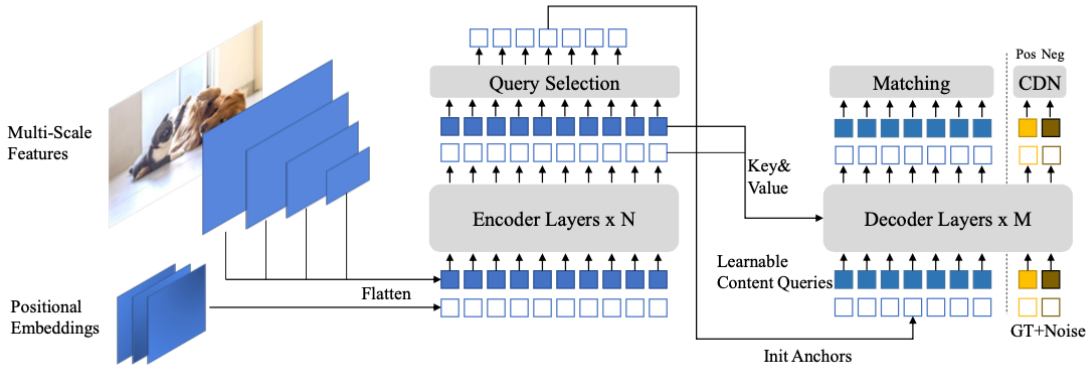
1. Architecture of the project detector

Framework of DINO model(DETR with improved DeNoising Anchor Boxes):



2. Implement details

A. Dynamic DN groups

As DETR-like models adopt mini-batch training, the total number of DN queries for each image in one batch is padded to the largest one in the batch. The design is inefficient and results in excessive memory consumption. So DINO propose to fix the number of DN queries and dynamically adjust the number of groups for each images according to its number of objects

B. Large-scale model pre-training.

There are two main different backbones used in DINO, one is ResNet-50 and the other is SwinL. I use SwinL pre-trained on Object365 and fine-tuned on train2017 as my pretrained model, Which is pre-trained DINO on Objects365 for 26 epochs using 64 Nvidia A100 GPUs and fine-tune the model on COCO for 18 epochs using 16 Nvidia A100 GPUS as showed in the DINO paper.

Pretrain download: https://drive.google.com/drive/folders/1qD5m1NmK0kjE5hh-G17XUX751WsEG-h

Swin backbone:
https://github.com/SwinTransformer/storage/releases/download/v1.0.0/swin_large_patch4_window12_384_22k.pth

C. hyper parameters

Item	Value
lr	0.0001
lr_backbone	1e-05
weight_decay	0.0001
clip_max_norm	0.1
pe_temperature	20
enc_layers	6
dec_layers	6
dim_feedforward	2048
hidden_dim	256
dropout	0.0
nheads	8
num_queries	900
enc_n_points	4
dec_n_points	4
transformer_activation	"relu"
batch_norm_type	"FrozenBatchNorm2d"
set_cost_class	2.0
set_cost_bbox	5.0
set_cost_giou	2.0
cls_loss_coef	1.0
bbox_loss_coef	5.0
giou_loss_coef	2.0
focal_alpha	0.25
dn_box_noise_scale	0.4
dn_label_noise_ratio	0.5

I use the default hyper parameter from the setting of DINO paper. But since my GPU is 30800 Ti, I can't train the model with batch size of 2, so I set the batch size to 1 and set the epoch to 100. I also set learning rate to 0.00005 in experiment, which is show in the next section.

D. Loss function

DINO use L1 loss and GIOU loss for box regression and focal loss with $\alpha = 0.25$, $\gamma = 2$ for classification. DINO add auxiliary losses after each decoder layer as DETR and add extra intermediate losses after the query selection module, with the same components as for each decoder layer. And loss coefficients are set to 1.0 for classification loss, 5.0 for L1 loss, and 2.0 for GIOU loss.

E. Augmentation

For the training augmentation, I use the method in DINO paper that randomly resize input image with its shorter side between 480 and 800 pixels and its longer side at most 1333. As for the pre-trained SwinL I used for training, I use default setting but finetune using 1.5 x larger scale (shorter side between 720 and 1200 pixels and longer side at most 2000 pixels), which is the same as the setting they use when they are training the model I used as pretrained model.

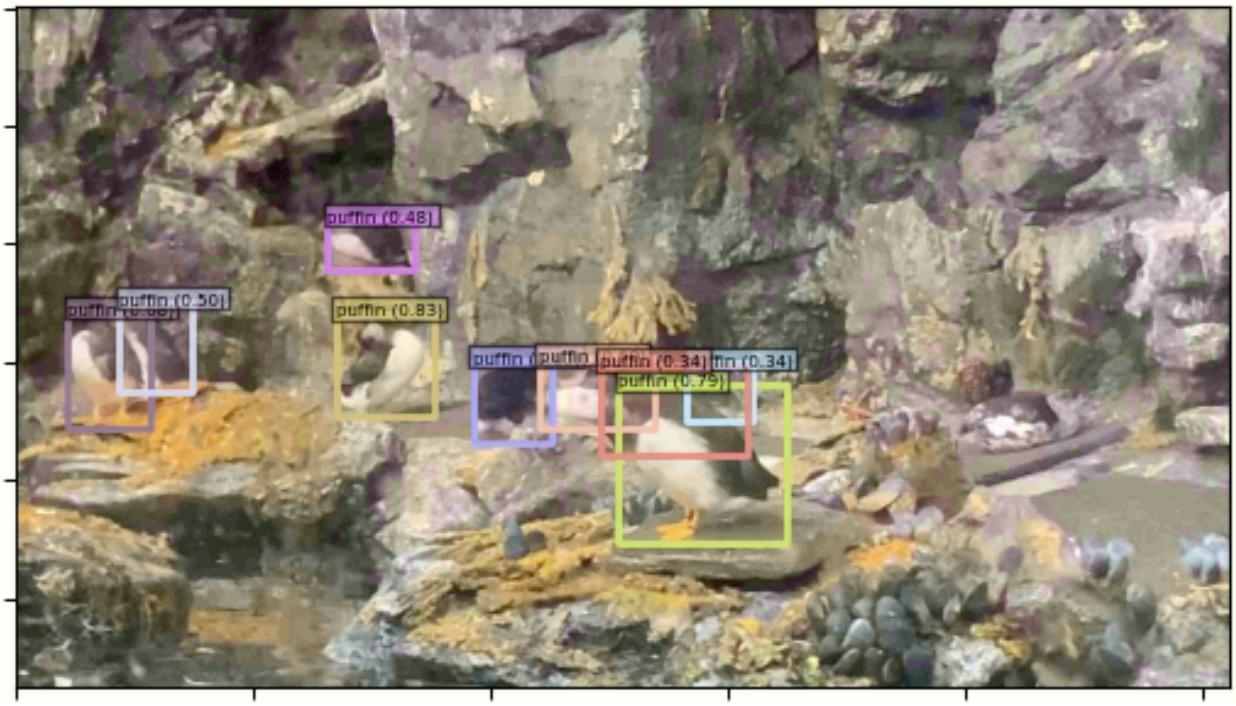
3. Performance of validation

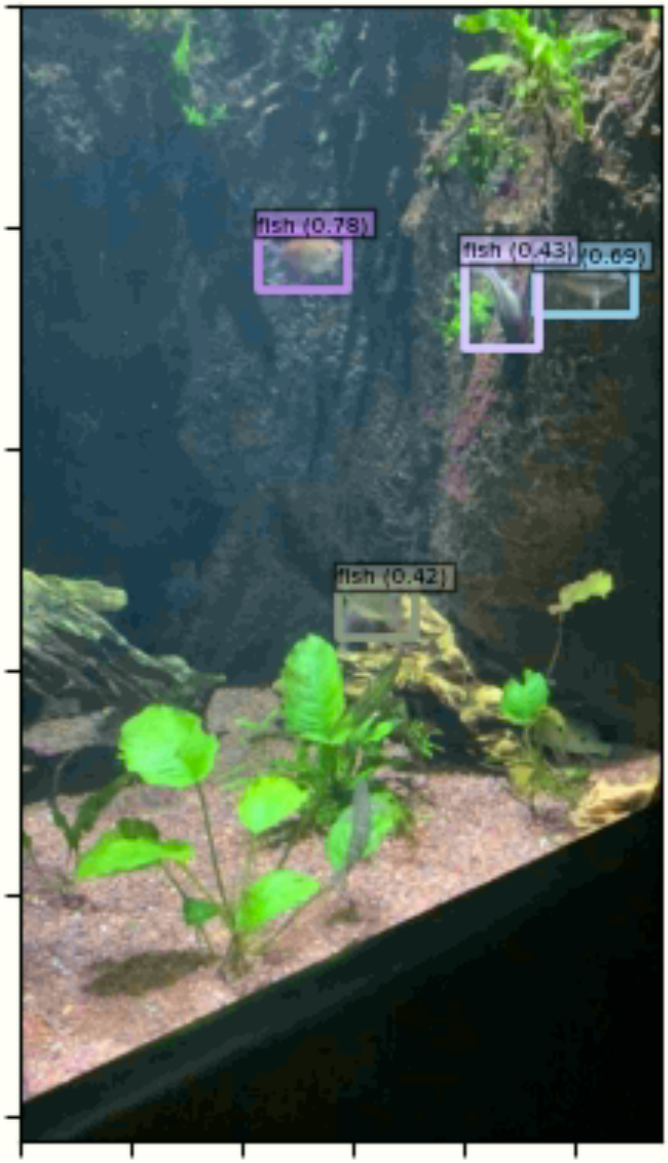
model	DINO_4scale_swin	DINO_4scale_swin	DINO_4scale
Learning rate	0.0001	0.00005	0.0001
epoch	100	100	100
AP	0.5852	0.5854	0.5140
AP50	0.8654	0.8629	0.7984
AP75	0.6133	0.6154	0.5301

I choose DINO_4scale_swin with 0.00005 learning rate and 100 epochs as my final result, which is trained with SwinL backbone and have the bast result for validation. The last column is the result of using DINO with Resnet50 as backbone.

Since the first epoch of DINO_4scale_swin with 0.0001 learning rate is already better than DINO_4scale with learning rate 0.0001 after 100 epochs, I didn't train it with 0.00005 learning rate.

4. visualization




fish (0.78)
fish (0.43)
(0.69)
fish (0.42)

5. reference

[1] https://arxiv.org/abs/2203.03605