Enhanced Training of Query-Based Object Detection via Selective Query Recollection

Fangyi Chen[1] Han Zhang[1] Kai Hu[1] Yu-Kai Huang[1] Chenchen Zhu[2] Marios Savvides[1]
Carnegie Mellon University[1] Meta AI[2]
{fangyic,hanz3,kaihu,yukaih2,marioss}@andrew.cmu.edu chenchenz@fb.com

Abstract

This paper investigates a phenomenon where query-based object detectors mispredict at the last decoding stage while predicting correctly at an intermediate stage. We review the training process and attribute the overlooked phenomenon to two limitations: lack of training emphasis and cascading errors from decoding sequence. We design and present Selective Query Recollection (SQR), a simple and effective training strategy for query-based object detectors. It cumulatively collects intermediate queries as decoding stages go deeper and selectively forwards the queries to the downstream stages aside from the sequential structure. Such-wise, SQR places training emphasis on later stages and allows later stages to work with intermediate queries from earlier stages directly. SQR can be easily plugged into various query-based object detectors and significantly enhances their performance while leaving the inference pipeline unchanged. As a result, we apply SQR on Adamixer, DAB-DETR, and Deformable-DETR across various settings (backbone, number of queries, schedule) and consistently brings $1.4 \sim 2.8$ *AP improvement. Code is available at* https://github.com/Fangyi-Chen/SQR

1. Introduction

Object detection is a long-established topic in computer vision aiming to localize and categorize objects of interest. Previous methods [4, 7, 10, 11, 16, 18, 21, 25, 26, 29, 32, 33, 35–37] rely on dense priors tiled at feature grids so as to detect in a sliding-window paradigm, and have dominated object detection for the recent decade, but these methods fail to shake off many hand-crafted processing steps such as anchor generation or non-maximum suppression, which block end-to-end optimization.

Recent research attention has been geared towards query-based object detection [3, 17, 20, 23, 28, 31, 38] since the thriving of transformer [30] and DETR [3]. By viewing detection as a direct set prediction problem, the new archetype represents the set of objects using a set of learnable embeddings, termed as queries, which are fed to a decoder consisting of a stack (typically six) of decoding stages. Each stage performs similar operations: (1) interacting queries with image features via an attention-like mechanism, so the queries are aggregated with valuable information that represents objects; (2) reasoning the relation among all queries so that global dependency on objects co-occurrence and duplicates could be captured; (3) interpreting bounding box and category from each query by a feed forward network. Queries are sequentially processed stage-by-stage, and each stage is formulated to learn a residual function with reference to the former stage's output, aiming to refine queries in a cascaded style.

As such wise, the decoding procedure implies that detection should be stage-by-stage enhanced in terms of IoU and confidence score. Indeed, monotonically improved AP is empirically achieved by this procedure. However, when visualizing the stage-wise predictions, we surprisingly observe that decoder makes mistakes in a decent proportion of

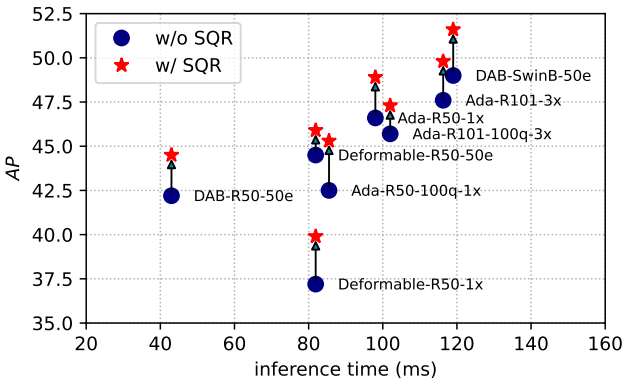

Figure 1. The inference speed and AP for various networks on the MS-COCO val set. The red stars are the results trained with SQR. The blue circles are the results of baselines without SQR. SQR enhances the training of query-based object detectors while leaving the inference pipeline unchanged.

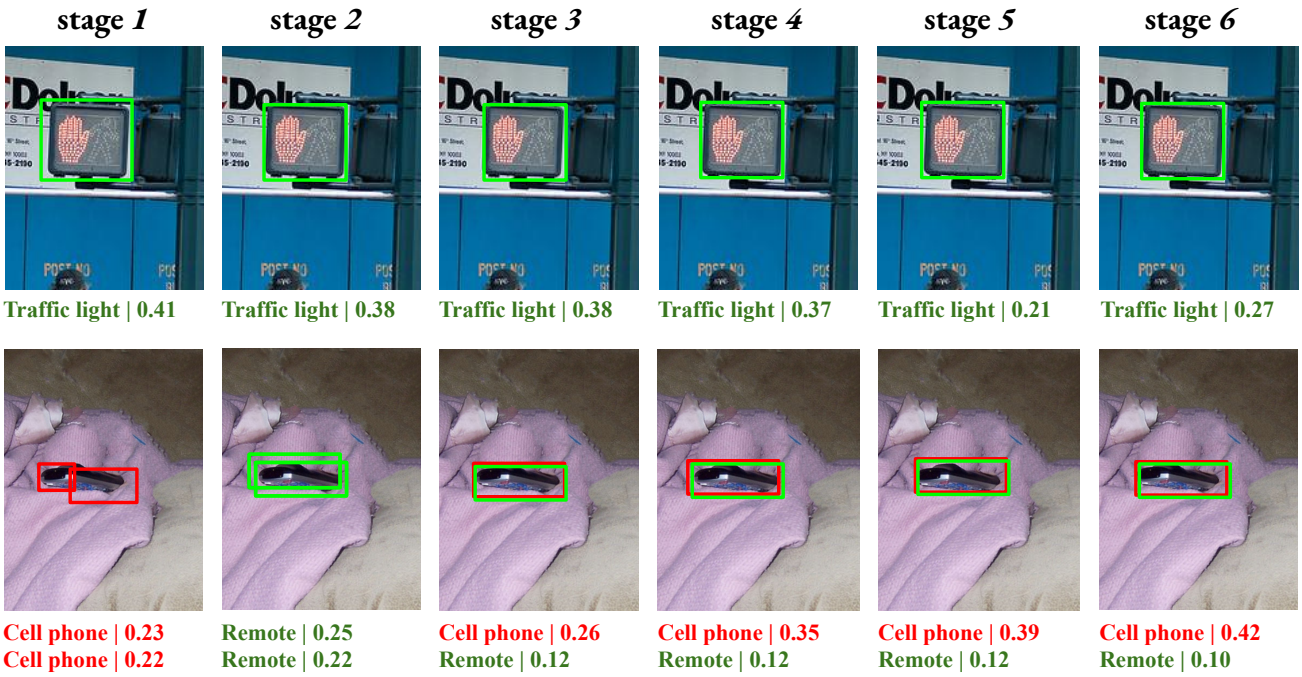

Figure 2. Are query-based object detectors always enhancing predictions stage-by-stage? The *traffic light* at stage 1 gets a confident score of 0.41, while from stage 2 to 5 the confidence gradually decreases to 0.21 (Upper); the *remote* at stage 3 was wrongly classified as a cell phone, and from stage 3 to 6 the mistake was amplified from 0.26 to 0.42 (Lower). The visualization is acquired from Adamixer-R50 (42.5 AP) tested on COCO val set.

cases where the later stages degrade true-positives and upgrade false-positives from the former stages. As shown in Fig.2, the traffic light at stage 1 gets categorical confidence of 0.41, while from stage 2 to 5 the confidence gradually decreases to 0.21; the remote at stage 3 was wrongly classified as a cell phone, while from stage 3 to 6 the error was exacerbated from 0.26 to 0.42. We present a more detailed statistic in Section 3.

This phenomenon inspires us to review the current training strategy and bring two conjectures. **Firstly**, the responsibility that each stage takes is unbalanced, while supervision applied to them is analogous. An early stage could make mistakes without causing too much impact because it gets chances to be corrected later, and the later stages are more responsible for the final prediction. But during training, all of these stages are supervised in an equivalent manner and there lacks such a mechanism that places particular training emphasis on later stages. **Secondly**, due to the sequential structure of the decoder, an intermediate query refined by a stage - no matter whether this refinement brings positive or negative effects - will be cascaded to the following stages, while the query prior to the refinement never gets an opportunity to be propagated forward even though it emerges unscathed and might be more representative than the refined one. The cascading errors increase the difficulty of convergence and the sequential structure impedes the later stages from seeing prior queries during training.

Based on these intuitions, we present Query Recollection (QR) as a training strategy for query-based object detectors. It cumulatively collects intermediate queries as stages go deeper, and feeds the collected queries to the downstream stages aside from the sequential structure. By each stage, the new add-ins alongside the original inputs are independently treated among each other, so the attentions and losses are calculated individually. In such a manner, QR enjoys two key features: **(1)** The number of supervision signals per stage grows in geometric progression, so that later stages get more supervision than the former ones, for example, the sixth stage got 32 times more supervision than the first; **(2)** Later stages get chance to view the outputs beyond its neighboring stage for training, which mitigates the potential impact due to cascading errors. We further discover that *selectively* forward queries to each stage, not with the entire query collection but only those from the prior two stages, can raise the number of supervision in a Fibonacci sequence which halves the extra computing cost and brings even better results. We name it Selective Query Recollection (SQR).

Our contributions are summarized in three folds: **(1)** We quantitatively investigate the phenomenon where query-based object detectors mispredict at the last decoding stage

while predicting correctly at an intermediate one. **(2)** We attribute the overlooked phenomenon to two training limitations, and propose a simple and effective training strategy SQR that elegantly fits query-based object detectors. **(3)** We conduct experiments on Adamixer, DAB DETR, and Deformable DETR across various training settings that verify its effectiveness (Fig.1).

2. Related Work

2.1. Training Strategy for Object Detection

Detectors based on dense priors have been dominating the community for decades. The abstract concept anchor box or anchor point [26, 29] aims to match with ground truth (GT) objects depending on their Intersection-over-Union (IoU) values or other advanced soft scoring factors [7, 15, 18, 29, 37]. Among anchor-based detectors, multi-stage models iteratively refine bounding box and category stage-by-stage. A typical example is Cascade RCNN [2] which is based on the design that the output of an intermediate stage is sampled and re-labeled to train the next stage with increasing IoU thresholds, so these stages are guaranteed to be progressively refined. Recently, DETR [3] starts a family of end-to-end query-based models where object detection is regarded as a set prediction problem. To train DETR, a predefined number of object queries are matched to either a ground-truth or background by solving the Hungarian Matching problem. The queries are refined by several decoder stages similar to Cascade RCNN, and each intermediate stage is supervised by the matching results.

2.2. Query-Based Object Detection

Recently, many algorithms have been following the idea of DETR. Deformable DETR [38] proposes a deformable attention module that alleviates the aforementioned issues and massively improves the convergence speed by a factor of 10. Conditional DETR [23] decouples the object query into content query and spatial query in the decoder cross-attention module and learns a conditional spatial query from the decoder embedding to enable fast learning of the distinctive extremity of the ground-truth objects. Anchor-DETR [31] formulates the object queries as anchor points such that each object query may only focus on a certain region near its anchor point. Many future works are inspired by this design. DAB-DETR [20] dives deep into the role of object queries. It directly uses anchor box coordinates as spatial query to speed up training. The model benefits from the spatial prior by modulating the positional attention map using the width and height of the box. DN-DETR [17] further improves the convergence speed and query matching stability of DAB-DETR with the help of the Ground Truth denoising task. Adamixer [9] re-designs the query-key pooling mechanism by letting the query adaptively attend to the meaningful regions of the encoded features directly through bilinear sampling. An MLP-Mixer then dynamically decodes the pooled features into final predictions. DETA (Detection Transformers with Assignment) [24] proposes a novel overlap-based assignment that enables one-to-many assignments to DETR.

Despite their significant improvement on DETR, they focus little on the training emphasis and cascading errors. We propose SQR to pay attention to the two problems.

Model	S1	S2	S3	S4	S5	S6
Deformable	38.4	42.2	43.7	44.2	44.4	44.5
Adamixer	15.1	30.3	37.7	40.6	42.1	42.5

Table 1. Stage-wise AP results. Deformable DETR(300 queries) and Adamixer(100 queries) are trained using official implementation, both of their decoders consist of 6 stages, and per stage AP is reported on COCO val set.

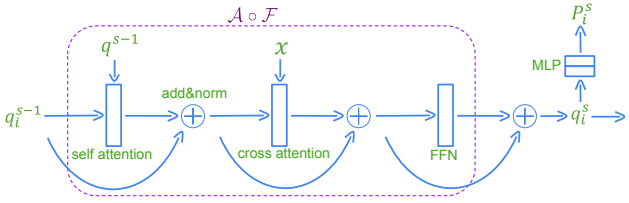

Figure 3. Typical structure of a single decoding stage.

3. Motivation

Are query-based object detectors always predicting the optimal detections at the last stage? Table.1 shows that AP gradually increases with stages going deeper, indicating improved predictions on a general scale. While the observation in Fig.2 implies that simple AP results are insufficient for an in-depth analysis of that question.

Preliminary. Queries are updated successively. A typical structure of a decoding stage is illustrated in Fig.3. An initial query q_i^0 ($i \in N = \{1, 2, ..., n\}$) is an embedding trained through back-propagation, and n is the total number of initial queries. During inference, the first stage updates the initial query by adding a residual term to it, producing intermediate query q_i^1, followed by later stages sequentially updating the intermediate query in the same way. The procedure can be formulated as

$$q_i^s = D^s(q_i^{s-1}, q^{s-1}, x) = q_i^{s-1} + (\mathcal{A} \circ \mathcal{F})(q_i^{s-1}, q^{s-1}, x) \tag{1}$$

D^s is a decoding stage where s is stage index; q_i^s is the i_{th} query at stage s; q^s is a set of queries $q^s = \{q_i^s | i \in N\}$; $(\mathcal{A} \circ \mathcal{F})$ stands for a bundle of modules including self and cross attention and feed forward network; x means features; and LayerNorm [1] that applied on each module is omitted

Model	TP Threshold	TP F Rate	FP E Rate
Deformable DETR	IoU>0.50	51.4%	55.7%
	IoU>0.75	49.5%	55.9%
Adamixer	IoU>0.50	28.6%	50.8%
	IoU>0.75	26.7%	51.2%

Table 2. True-positive Fading Rate and False-positive Exacerbation Rate.

for simplicity. Afterward, q_i^s predicts an object P_i^s via two multi-layer perceptrons for classification and regression:

$$P_i^s = (\ MLP_{cls}(q_i^s),\ \ MLP_{reg}(q_i^s)\ ) \tag{2}$$

$P_i^{1\sim6}$ are predicted by the $q_i^{1\sim6}$ rooted in q_i^0, where P_i^6 is the expected outcome and $P_i^{1\sim5}$ are intermediate results. P_i^s is regarded as a true-positive towards a ground-truth G only if the IoU(P_i^s, G) exceeds a threshold, its category matches with G, and the categorical score is ranked as the highest among all other counterparts.

Investigation. We study the stage-wise testing results by first defining two rates: (1) If P_i^6 is a true-positive (TP) towards a ground-truth G, we check whether $P_i^{1\sim5}$ generate a better TP ***towards the same G that has higher IoU*** & ***higher category score*** than P_i^6. The occurrence rate is denoted as *TP fading rate*. (2) If P_i^6 is a false-positive (FP), we check whether $P_i^{1\sim5}$ produce a FP ***but with lower category score*** than P_i^6. The occurrence rate is denoted as *FP exacerbation rate*.

The statistics are shown in Table.2. We can see that TP fading rate achieves 50% and 27% on the two models. Considering the strict constraints it holds, the TP fading rate reaches an impressively high level. Deformable DETR is much higher than Adamixer due to the smaller AP gap among stages, thus, in many cases, earlier stages are more likely to outperform the last stage. FP exacerbation holds looser constraints and is easier to be satisfied, its rate is above 50% on the two models. Higher TP threshold implies higher quality of P_i^6 and it is harder to find qualifiers in $P_i^{1\sim5}$, but the two rates are similar with 0.75 as with 0.5, pointing out the consistency of this phenomenon. We also observe that more than half of the occurred cases are *marginally triggered*, i.e. the predictions from the triggered $P_i^{1\sim5}$ are only marginally better than the sixth. This is a further reason why the deformable DETR has those high rates - the results from the 5th and 6th stages are extremely visually close. In dominated number of cases, the final stage is (one of) the best, after all, its mAP is the highest.

When the conditions of the two rates establish, we further replace P_i^6 with the optimal prediction found in $P_i^{1\sim6}$ and measure the AP. On Deformable DETR, AP grows from **44.5 AP** to **51.7 (+7.2 AP)**; on Adamixer, AP grows from **42.5 AP** to **53.3 (+10.7 AP)**, demonstrating huge potential in query-based detectors yet to be mined.

Conclusion. This reveals that models frequently predict the optimum at intermediate stages instead of the last one. We view the problem from the training's perspective and identify *the lack of training emphasis* and *the cascading errors from query sequence* as two obstacles impeding the occurrence of the most reliable predictions in the last stage, elaborated in Section 1.

4. Query Recollection

4.1. Expectancy

We desire such a training strategy that embraces:

- Uneven supervision applied to decoding stages that places emphasis on later ones, enhancing later stages for better final outcomes.
- A variety of early-stage queries directly introduced to later stages, mitigating the impact of cascading errors.

To this end, we design a concise training strategy coined as Query Recollection (QR). Compared with prior arts, it collects intermediate queries at every stage and forwards them along the original pathway. Dense Query Recollection (DQR) is the fundamental form and Selective Query Recollection (SQR) is an advanced variant.

4.2. Dense Query Recollection

Notation. The process of single decoding stage (self-attention, cross-attention, FFN), the ground-truth assignment, and loss calculations are applied within a set of queries $\{q_i | i \in \{1, 2, ..., n\}\}$, where n is typically 100/300/500. We regard the set of queries as a basic unit in our method and generally denote as q.

Basic Pathway. Query along the basic pathway is refined by all stages. We illustrate it in Fig.4(a). Taking a 4-stages decoder as an example, we denote $q^{0\text{-}1\text{-}2\text{-}3\text{-}4}$ as the final query that is refined by all stages. So the basic $\mathcal{PT}$ finally produces

$$q^{0\text{-}1\text{-}2\text{-}3\text{-}4} = D^4(D^3(D^2(D^1(q^0)))) \tag{3}$$
$$= \mathcal{PT}^{1\text{-}2\text{-}3\text{-}4}(q^0) \tag{4}$$

During training, the queries from each stage, i.e. $q^{0\text{-}1}$, $q^{0\text{-}1\text{-}2}$, $q^{0\text{-}1\text{-}2\text{-}3}$, and $q^{0\text{-}1\text{-}2\text{-}3\text{-}4}$ are independently followed by Hungarian Assignment that matches ground-truth with q in a one-to-one manner, and then followed by loss calculation for supervision. In Fig.4, we mark those q that require supervision as $\hat{q}$. Along the basic pathway, the number of $\hat{q}$ at each stage is 1.

DQR Formulation. We densely collect every intermediate q and independently forward them to every downstream

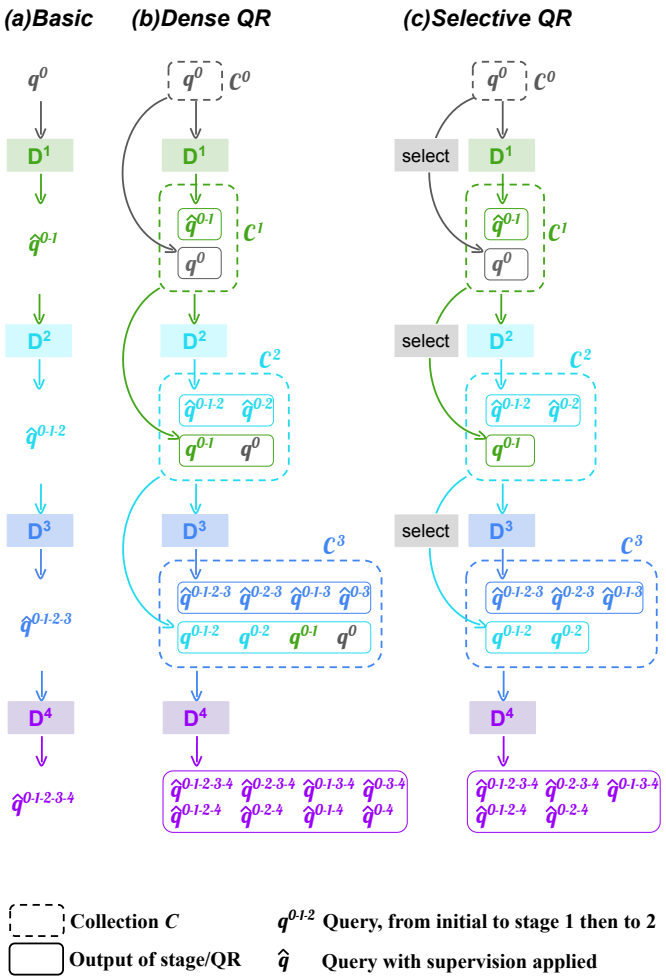

Figure 4. (a). Basic process for decoding queries stage by stage, applied in both training and testing. (b). Dense query recollection. (c). Selective query recollection.

stage, as illustrated in Fig.4(b). After each stage, a collection C is formed where the number of q grows geometrically, namely 2^s at s_{th} stage. Formally,

$$C^0 = \{q^0\} \tag{5}$$

$$C^s = \{D^s(q)|q \in C^{s-1}\} \cup C^{s-1} \tag{6}$$

In a collection C, half of queries inside are newly generated by the current stage, i.e. from $\{D^s(q)|q \in C^{s-1}\}$, while another half are from previous stages, C^{s-1}. Separately, for each q in the former half, we apply Hungarian assignment and loss calculation, so the number of supervision signals grows in geometric progression as well, namely 2^{s-1} at s_{th} stage.

Such-wise, Dense Query Recollection satisfies our expectancy where the number of supervision signal for each stage grows as (1,2,4,8,16,32), meanwhile, all prior queries would be visible in all later stages.

stage	1	2	3	4	5
TP F Rate(%)	1.2	4.4	8.5	12.4	16.9
FP E Rate(%)	14.3	18.6	20.8	24.5	30.2
stage	1∼3	4&5	3∼5	2∼5	1∼5
TP F Rate(%)	11.2	23.9	26.9	28.3	28.6
FP E Rate(%)	32.4	40.8	45.3	48.5	50.8

Table 3. Stage-wise TP Fading Rate and FP Exacerbation Rate with Adamixer.

During inference, we only use the basic pathway, so the inference process is untouched. For a standard 6-stage decoder, the pathway is $\mathcal{PT}^{1\text{-}2\text{-}3\text{-}4\text{-}5\text{-}6}$

4.3. Selective Query Recollection

The last proposal empirically enhances training, but the query collection process is indiscriminate, which brings two defects: First, the geometrical growth of the number of $\hat{q}$ and their attention/loss calculations cost a lot. Second, if we input an early q that skips too many stages to a far-away late stage, the potential benefit could be overshadowed by the huge learning gap between the stages and query. For instance, if the initial q^0 is directly introduced to stage 6 and produces $\hat{q}^{0\text{-}6}$, this query would have the highest difficulty among all queries at stage 6 and the calculated loss would dominate the overall losses. So we are inclined to selectively collect intermediate q rather than densely collect all of them.

Selection. To find a better query recollection scheme, we further conduct a detailed analysis on the *TP Fading Rate* and *FP Exacerbation Rate* introduced in Section 3.

Recall the spirits of the two rates where we seek from $P_i^{1\sim5}$ for an alternative that is better than P_i^6 , we want to investigate which specific intermediate stage/stages contribute the most. Concretely, if P_i^6 is a true-positive towards a ground-truth G, we separately check each stage whether generating a better TP; similarly, if P_i^6 is a false-positive, we separately check each stage whether generating a better FP. The results are summarized in Table 3. We find that the majority of alternatives of P_i^6 are from stage 4&5, where the corresponding TP fading rate and FP exacerbation rate reach 23.9% and 40.8%, respectively, which are close to the results of stages $1 \sim 5$. While stage $1 \sim 3$ together only produces 11.2% and 32.4%.

The above analysis implies that the queries from *the adjacent stage* and *the stage prior to the adjacent stage* are more likely to bring positive effects. We intuitively follow the observations and selectively operate collection along the basic pathway: before starting stage D^s, we collect q from the 2 nearest stages, i.e. D^{s-1} and D^{s-2} as the input of D^s.

SQR Formulation. The Selective Query Recollection

can be formulated as

$$C^0 = \{q^0\} \qquad C^1 = \{q^0, q^{0\text{-}1}\} \tag{7}$$

$$C^s = \{D^s(q)|q \in C^{s-1}\} \cup select(C^{s-1}) \tag{8}$$

$$= \{D^s(q)|q \in C^{s-1}\} \cup \{D^{s-1}(q)|q \in C^{s-2}\} \tag{9}$$

Such wise, Selective Query Recollection (Fig.4(c)) still satisfies our expectancy, and the number of supervision signals grows in a Fibonacci sequence (1,2,3,5,8,13). Compared to the Dense Recollection, to a great extent, SQR reduces the computing burden and we observe that SQR even outperforms the dense counterpart in terms of precision. This verifies our assumption that a q skipping too many stages might be noise for remote stages overshadowing its positive effects.

Recollection Starting Stage. Above we collect queries starting from stage 1. Instead, we can practically vary this starting stage, and this will further reduce the total queries in each collection and further reduce the computing burden. E.g., if starts SQR from stage 2, the Fibonacci sequence will start from stage 2 and result in (1,1,2,3,5,8); if starts from stage 3, result in (1,1,1,2,3,5). The starting stage is regarded as a hyper-parameter for SQR.

5. Experiments

We conduct our experiments on the MS-COCO [19] detection track using the MMDetection [5] and Detrex [27] code-bases. All models are trained on **train2017** split. Unless otherwise specified, models are trained and tested with image scale of 800 pixels, where AdamW optimizer with a standard 1x schedule (12 epochs) is utilized for training. For ablation study and analysis, Adamixer [9] with R50 [12] is chosen because of its good performance and fast convergence speed (42.5 AP with 1x schedule). We further apply SQR on other detectors to verify its effectiveness.

5.1. Ablation Study

Baseline vs DQR vs SQR. Table.4 shows that both DQR and SQR improve the baseline by a large margin. DQR reaches 44.2 (+1.7 AP) while SQR reaches a slightly high result 44.4 (+1.9 AP). Note that SQR is much more efficient than DQR. As shown in Table.5, under the same training setting, SQR cuts down a great amount of training time of DQR and still achieves equal or higher AP.

Varying Starting Stage of SQR. We present how SQR performs when varying the starting stage in Table.5. The best performance is acquired when query recollection starts at stage 1 but with the most computational cost. We can see that starting at stage 2 performs similarly to starting at stage 1 while the computing burden is decently reduced. With recollection starting later, the benefits from SQR decrease as expected since the recollected queries from early stages become fewer and training emphasis gets gradually balanced.

Methods	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
Baseline	42.5	61.5	45.6	24.6	45.1	59.2
DQR	44.2	62.8	47.9	26.7	46.9	60.5
SQR	44.4	63.2	47.8	25.7	47.4	60.2

Table 4. AP comparison among Baseline, DQR, and SQR

Method	Start Stage	Train Time	AP	AP_{50}
Baseline	-	1x(5hours)	42.5	61.5
Baseline	-	2x	42.5	61.3
Baseline	-	3x	42.5	61.4
DQR	-	2.24x	44.2	62.8
SQR	1	1.57x	44.4	63.2
SQR	2	1.34x	44.2	63.0
SQR	3	1.18x	43.8	62.3
SQR	4	1.07x	42.9	61.4

Table 5. Further comparison among Baseline, DQR, and SQR with different starting stage in terms of training time and AP.

Method	TP Threshold	TP F Rate	FP E Rate
Baseline	IoU>0.50	28.6%	50.8%
SQR	IoU>0.50	23.3 %	47.3 %
Baseline	IoU>0.75	26.7%	51.2%
SQR	IoU>0.75	21.1%	47.0%

Table 6. Baseline vs. SQR on true-positive fading rate and false-positive exacerbation rate.

TP Fading Rate And FP Exacerbation Rate. We present Table.6 to verify that TP fading rate and FP exacerbation rate decrease due to the training effect when applied SQR. Specifically, TP fading rate decreases from 28.6% to 23.3% and from 26.7% to 21.1% across two IoU thresholds. FP exacerbation rate downgrades from 50.8% to 47.3 and 51.2% to 47.0%.

5.2. Relation with Increased Number of Supervision

Some concurrent studies reveal that query-based detectors suffer from training inefficiency due to the one-to-one matching, and propose to add extra parallel query groups and match them with ground-truths in a one-to-many manner, such as Group DETR [6] and H-DETR [14]. We recognize them as *increased number of supervision*, i.e., each ground-truth is matched to multiple queries, and thus the number of supervision ($\hat{q}$) is greatly increased. However, simply adding more query groups is a sub-optimal solution according to our motivation, since these extra supervisions

Design	#$Supv$ / stage	#$Supv$	AP
I (Group DETR)	3,3,3,3,3,3	18	43.4
II	4,4,4,3,2,1	18	43.0
III	1,2,3,4,4,4	18	43.7
IV (SQR)	1,1,2,3,5,8	20	44.2
V (Group DETR)	6,6,6,6,6,6	36	43.6
VI (SQR)	1,2,3,5,8,13	32	44.4

Table 7. Results of the 6 designed training strategies on Adamixer to investigate the relation with number of supervision. The inference is untouched. #$Supv$ denotes the number of supervision.

are given uniformly among stages. So we present 6 designs as training strategies for investigation.

- **Design I**: Following Group DETR, 3 groups of queries are initialized and independently undergo the whole process of the decoder. The pathway is 3 $\times$ $\mathcal{PT}^{\text{1-2-3-4-5-6}}$, and number of supervision is 3 groups $\times$6 stages = 18.
- **Design II**: 4 groups of queries are initialized but the pathways are $\mathcal{PT}^{\text{1-2-3-4-5-6}}$, $\mathcal{PT}^{\text{1-2-3-4-5}}$, $\mathcal{PT}^{\text{1-2-3-4}}$, $\mathcal{PT}^{\text{1-2-3}}$, so, with training emphasis on early stages, the number of supervision is $4+4+4+3+2+1 = 18$.
- **Design III**: Similar to Design II but not every initialized queries starts from stage 1. The pathways for the 4 groups are $\mathcal{PT}^{\text{1-2-3-4-5-6}}$, $\mathcal{PT}^{\text{2-3-4-5-6}}$, $\mathcal{PT}^{\text{3-4-5-6}}$, $\mathcal{PT}^{\text{4-5-6}}$, so, with training emphasis on later stages, the number of supervision is $1+2+3+4+4+4 = 18$.
- **Design IV**: $SQR^{starting\ stage=2}$, the number of supervision is $1+1+2+3+5+8 = 20$.
- **Design V**: Same as Design I but with 6 groups. The number of supervision is 6 groups $\times$6 stages = 36.
- **Design VI**: $SQR^{starting\ stage=1}$, the number of supervision is $1+2+3+5+8+13 = 32$.

Table.7 summarizes the results. From the controlled experiments with Design I, II, III, we conclude training with emphasis on late stages is the best option. Design IV and VI are SQR with different starting stages. We show that with similar or even fewer supervisions, SQR outperforms other designs by a large margin. These comparisons verify our motivation and prove that the enhancement from SQR is not simply from the increased number of supervisions.

5.3. Training Emphasis via Re-weighted Loss

Although *the training emphasis* contributes to the success of SQR, the *access to the early intermediate queries* is irreplaceable. We attempt another way to place training emphasis without query recollection, i.e. re-weighting the losses of six decoding stages following the Fibonacci sequence (1,2,3,5,8,13). The AP slightly degrades by (-0.6) compared to the baseline, indicating access to the intermediate queries is necessary for emphasized supervision.

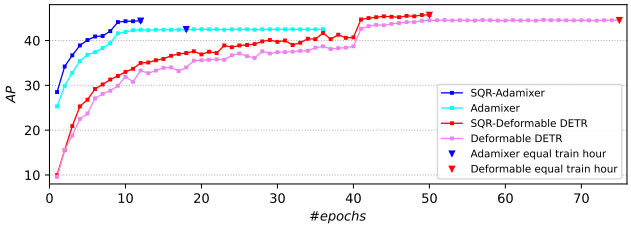

Figure 5. Equal training-time comparison: Baseline vs. SQR

5.4. Relation with Stochastic Depth

Stochastic Depth (SD) [13] is a training strategy for very deep network where it randomly removes a fraction of layers independently for each mini-batch, so the depth of pathway varies during training just like SQR. However, SD is of vital inefficiency, because an image has to first pass through a backbone and then be processed by a varying-depth pathway, but for SQR, an image that passes through the backbone can be processed by multiple varying-depth pathways at the same time. To verify this point, we apply SD on the vanilla decoder: During training, for each mini-batch we randomly remove decoding stages following a series of predefined probabilities. From stage 1$\sim$ 6, the removal probability $\mathbb{R}$ could be either a constant (0.1), increasing (0.0, 0.1,..., 0.5), or decreasing (0.5, 0.4,..., 0.0). During testing, SD requires the outputs of each decoding stage to be calibrated by the expected times the stage participates in training, so we multiply (1-$\mathbb{R}$) with the residual term in Equ.1. As a result, SD (40.7 mAP) is not comparable to SQR.

5.5. Training efficiency

SQR leads to an extra computing burden compared to the baseline since the number of $\hat{q}$ grows. In Table.5, SQR increases training time of Adamixer-R50 by 0.35$\sim$2.85 hours ($+0.07\% \sim 57\%$). In practice, the effects on different models/implementations/platforms vary. For instance, ResNet-50 is a light-weighted backbone, so the effects of applying SQR are relatively high, while with heavier backbones like ResNet-101 and Swin Transformer [22], the training time is less impacted (+10%).

For comparison under equal training time, we train the baseline Adamixer with extra epochs (+ 200% epochs) and baseline Deformable DETR with (+50% epochs). Both models' performances (see Fig.5) saturate early and there is no significant improvement over the extended training time, indicating that the benefit of SQR cannot be compensated by including more training iterations.

5.6. Comparison with State-of-the-art

We conduct experiments on recent query-based detectors with different training settings and various backbones, with and without SQR. We primarily follow the original training

Model	w/ SQR	#query	#epochs	COCO 2017 validation split					
				AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
DETR-R50 [3]		100	500	42.0	62.4	44.2	20.5	45.8	61.1
Conditional DETR-R50 [23]		300	50	40.9	61.8	43.3	20.8	44.6	60.2
Conditional DETR-R101 [23]		300	50	42.8	63.7	46.0	21.7	46.6	60.9
Anchor-DETR-R50 [31]		300	50	42.1	63.1	44.9	22.3	46.2	60.0
Anchor-DETR-R101 [31]		300	50	43.5	64.3	46.6	23.2	47.7	61.4
SAM-DETR-R50 [34]		300	50	39.8	61.8	41.6	20.5	43.4	59.6
*SMCA-DETR-R50 [8]		300	50	43.7	63.6	47.2	24.2	47.0	60.4
*SMCA-DETR-R50 [8]		300	108	45.6	65.5	49.1	25.9	49.3	62.6
*DN-DAB-DETR-R50 [17]		300	50	44.1	64.4	46.7	22.9	48.0	63.4
*DN-DAB-DETR-R101 [17]		300	50	45.2	65.5	48.3	24.1	49.1	65.1
*DAB-DETR-R50 [20]		300	50	42.2	63.1	44.7	21.5	45.7	60.3
*SQR-DAB-DETR-R50	✓	300	50	**44.5** (+2.3)	64.4	47.5	24.8	48.6	61.7
*DAB-DETR-SwinB [20]		300	50	49.0	71.0	53.0	29.6	53.8	68.3
*SQR-DAB-DETR-SwinB	✓	300	50	**51.6** (+2.6)	72.5	55.9	32.0	56.8	71.0
*Deformable DETR-R50 [38]		300	12	37.2	55.2	40.4	20.6	40.2	50.2
*SQR-Deformable DETR-R50	✓	300	12	**39.9** (+2.7)	58.4	43.7	23.8	43.2	53.3
*Deformable DETR-R50 [38]		300	50	44.5	63.2	48.9	28.0	47.8	58.8
*SQR-Deformable DETR-R50	✓	300	50	**45.9** (+1.4)	64.7	50.2	27.7	49.2	60.5
Adamixer-R50 [9]		100	12	42.5	61.5	45.6	24.6	45.1	59.2
SQR-Adamixer-R50	✓	100	12	**44.4** (+1.9)	63.2	47.8	25.7	47.4	60.2
†Adamixer-R50 [9]		100	12	42.5	61.5	45.8	24.4	45.2	58.7
†SQR-Adamixer-R50	✓	100	12	**45.3** (+2.8)	63.8	49.0	26.8	48.1	62.2
*†Adamixer-R50		100	36	45.1	63.9	48.9	28.3	47.8	60.6
*†SQR-Adamixer-R50	✓	100	36	**46.7** (+1.6)	65.2	50.3	29.4	49.6	62.1
*†Adamixer-R50		300	36	46.6	65.5	50.6	29.3	49.3	62.3
*†SQR-Adamixer-R50	✓	300	36	**48.9** (+2.3)	67.5	53.2	32.0	51.8	63.7
*†Adamixer-R101 [9]		100	36	45.7	64.7	49.6	27.8	49.1	61.2
*†SQR-Adamixer-R101	✓	100	36	**47.3** (+1.6)	66.0	51.3	30.1	50.7	62.2
*†Adamixer-R101 [9]		300	36	47.6	66.7	51.8	29.5	50.5	63.3
*†SQR-Adamixer-R101	✓	300	36	**49.8** (+2.2)	68.8	54.0	32.0	53.4	65.1

Table 8. Comparison results with various query-based detectors on COCO 2017 val. #query: the number of queries used during inference. * indicates models trained with multi-scale augmentation, † marks models with 7 decoder stages.

setting of our baselines, where training schedules consists of standard 12, 36, and 50 epochs; the number of queries is chosen between 100 and 300; multi-scale training is applied to 36e and 50e schedules as the shorter side of images range $480 \sim 800$. The training is conducted on 8x Nvidia A100.

The result is summarized in Fig.1 and Table 8. SQR consistently brings AP improvements to Adamixer, DAB-DETR and Deformable-DETR at the same inference speed. Concretely, on DAB-DETR, SQR brings +2.3 and +2.6 AP with R50 and SwinB, respectively; on Deformable DETR, SQR boosts it by 2.7 AP under 12e and by 1.4 AP under 50e; on Adamixer with R50, SQR achieves +1.9 AP under the basic setting (100 queries, 12e). With an additional stage, the gap between w/ and w/o SQR is enlarged +2.8 AP. SQR consistently improves these models by $1.4\sim2.8$ AP.

6. Conclusion

In this work, we investigate the phenomenon where the optimal detections of query-based object detectors are not always from the last decoding stage, but can sometimes come from an intermediate decoding stage. We first recognize two limitations causing the issue, i.e. lack of training emphasis and cascading errors from query sequence. The problem is addressed by Selective Query Recollection (SQR) as a simple and effective training strategy. Across various training settings, SQR boosts Adamixer, DAB-DETR, and Deformable-DETR by a large margin.

References

[1] Jimmy Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer normalization. *ArXiv*, abs/1607.06450, 2016. 3

[2] Zhaowei Cai and Nuno Vasconcelos. Cascade r-cnn: Delving into high quality object detection. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6154–6162, 2018. 3

[3] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. *ArXiv*, abs/2005.12872, 2020. 1, 3, 8

[4] Fangyi Chen, Chenchen Zhu, Zhiqiang Shen, Han Zhang, and M. Savvides. Ncms: Towards accurate anchor free object detection through l2 norm calibration and multi-feature selection. *Comput. Vis. Image Underst.*, 200:103050, 2020. 1

[5] Kai Chen, Jiaqi Wang, Jiangmiao Pang, Yuhang Cao, Yu Xiong, Xiaoxiao Li, Shuyang Sun, Wansen Feng, Ziwei Liu, Jiarui Xu, Zheng Zhang, Dazhi Cheng, Chenchen Zhu, Tianheng Cheng, Qijie Zhao, Buyu Li, Xin Lu, Rui Zhu, Yue Wu, Jifeng Dai, Jingdong Wang, Jianping Shi, Wanli Ouyang, Chen Change Loy, and Dahua Lin. MMDetection: Open mmlab detection toolbox and benchmark. *arXiv preprint arXiv:1906.07155*, 2019. 6

[6] Qiang Chen, Xiaokang Chen, Gang Zeng, and Jingdong Wang. Group detr: Fast training convergence with decoupled one-to-many label assignment. *ArXiv*, abs/2207.13085, 2022. 6

[7] Chengjian Feng, Yujie Zhong, Yu Gao, Matthew R Scott, and Weilin Huang. Tood: Task-aligned one-stage object detection. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 3490–3499. IEEE Computer Society, 2021. 1, 3

[8] Peng Gao, Minghang Zheng, Xiaogang Wang, Jifeng Dai, and Hongsheng Li. Fast convergence of detr with spatially modulated co-attention. *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 3601–3610, 2021. 8

[9] Ziteng Gao, Limin Wang, Bing Han, and Sheng Guo. Adamixer: A fast-converging query-based object detector. *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5354–5363, 2022. 3, 6, 8

[10] Ross B. Girshick. Fast r-cnn. *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 1440–1448, 2015. 1

[11] Ross B. Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 580–587, 2014. 1

[12] Kaiming He, X. Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016. 6

[13] Gao Huang, Yu Sun, Zhuang Liu, Daniel Sedra, and Kilian Q. Weinberger. Deep networks with stochastic depth. In *ECCV*, 2016. 7

[14] Ding Jia, Yuhui Yuan, Hao He, Xiao pei Wu, Haojun Yu, Weihong Lin, Lei huan Sun, Chao Zhang, and Hanhua Hu. Detrs with hybrid matching. *ArXiv*, abs/2207.13080, 2022. 6

[15] Kang jik Kim and Hee Seok Lee. Probabilistic anchor assignment with iou prediction for object detection. In *ECCV*, 2020. 3

[16] Tao Kong, Fuchun Sun, Huaping Liu, Yuning Jiang, and Jianbo Shi. Foveabox: Beyond anchor-based object detector. *ArXiv*, abs/1904.03797, 2019. 1

[17] Feng Li, Hao Zhang, Shi guang Liu, Jian Guo, Lionel M. Ni, and Lei Zhang. Dn-detr: Accelerate detr training by introducing query denoising. *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 13609–13617, 2022. 1, 3, 8

[18] Tsung-Yi Lin, Priya Goyal, Ross B. Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 2999–3007, 2017. 1, 3

[19] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft coco: Common objects in context. In *ECCV*, 2014. 6

[20] Shilong Liu, Feng Li, Hao Zhang, Xiao Bin Yang, Xianbiao Qi, Hang Su, Jun Zhu, and Lei Zhang. Dab-detr: Dynamic anchor boxes are better queries for detr. *ArXiv*, abs/2201.12329, 2022. 1, 3, 8

[21] W. Liu, Dragomir Anguelov, D. Erhan, Christian Szegedy, Scott E. Reed, Cheng-Yang Fu, and Alexander C. Berg. Ssd: Single shot multibox detector. In *ECCV*, 2016. 1

[22] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 9992–10002, 2021. 7

[23] Depu Meng, Xiaokang Chen, Zejia Fan, Gang Zeng, Houqiang Li, Yuhui Yuan, Lei Sun, and Jingdong Wang. Conditional detr for fast training convergence. *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 3631–3640, 2021. 1, 3, 8

[24] Jeffrey Ouyang-Zhang, Jang Hyun Cho, Xingyi Zhou, and Philipp Krähenbühl. Nms strikes back, 2022. 3

[25] Joseph Redmon, Santosh Kumar Divvala, Ross B. Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 779–788, 2016. 1

[26] Shaoqing Ren, Kaiming He, Ross B. Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39:1137–1149, 2015. 1, 3

[27] Tianhe Ren, Shilong Liu, Hao Zhang, Feng Li, Xingyu Liao, and Lei Zhang. detrex. https://github.com/IDEA-Research/detrex, 2022. 6

[28] Pei Sun, Rufeng Zhang, Yi Jiang, Tao Kong, Chenfeng Xu, Wei Zhan, Masayoshi Tomizuka, Lei Li, Zehuan Yuan, Changhu Wang, and Ping Luo. Sparse r-cnn: End-to-end object detection with learnable proposals. *2021 IEEE/CVF*

Conference on Computer Vision and Pattern Recognition (CVPR), pages 14449–14458, 2021. 1

[29] Zhi Tian, Chunhua Shen, Hao Chen, and Tong He. Fcos: Fully convolutional one-stage object detection. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 9626–9635, 2019. 1, 3

[30] Ashish Vaswani, Noam M. Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *ArXiv*, abs/1706.03762, 2017. 1

[31] Yingming Wang, X. Zhang, Tong Yang, and Jian Sun. Anchor detr: Query design for transformer-based detector. In *AAAI*, 2022. 1, 3, 8

[32] Haibao Yu, Qi Han, Jianbo Li, Jianping Shi, Guangliang Cheng, and Bin Fan. Search what you want: Barrier panelty nas for mixed precision quantization. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *Computer Vision – ECCV 2020*, pages 1–16, Cham, 2020. Springer International Publishing. 1

[33] Haibao Yu, Yizhen Luo, Mao Shu, Yiyi Huo, Zebang Yang, Yifeng Shi, Zhenglong Guo, Hanyu Li, Xing Hu, Jirui Yuan, and Zaiqing Nie. Dair-v2x: A large-scale dataset for vehicle-infrastructure cooperative 3d object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 21361–21370, June 2022. 1

[34] Gongjie Zhang, Zhipeng Luo, Yingchen Yu, Kaiwen Cui, and Shijian Lu. Accelerating detr convergence via semantic-aligned matching. *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 939–948, 2022. 8

[35] Haoyang Zhang, Ying Wang, Feras Dayoub, and Niko Sunderhauf. Varifocalnet: An iou-aware dense object detector. *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8510–8519, 2021. 1

[36] Shifeng Zhang, Cheng Chi, Yongqiang Yao, Zhen Lei, and Stan Z. Li. Bridging the gap between anchor-based and anchor-free detection via adaptive training sample selection. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9756–9765, 2020. 1

[37] Chenchen Zhu, Fangyi Chen, Zhiqiang Shen, and Marios Savvides. Soft anchor-point object detection. In *ECCV*, 2020. 1, 3

[38] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable detr: Deformable transformers for end-to-end object detection. *ArXiv*, abs/2010.04159, 2021. 1, 3, 8