DetKDS: Knowledge Distillation Search for Object Detectors

Lujun Li [1] **Yufan Bao** [1] **Peijie Dong** [2] **Chuanguang Yang** [3] **Anggeng Li** [1] **Wenhan Luo** [1] **Qifeng Liu** [1] **Wei Xue** [1] **Yike Guo** [1]

Abstract

In this paper, we present DetKDS, the first framework that searches for optimal detection distillation policies. Manual design of detection distillers becomes challenging and time-consuming due to significant disparities in distillation behaviors between detectors with different backbones, paradigms, and label assignments. To tackle these challenges, we leverage search algorithms to discover optimal distillers for homogeneous and heterogeneous student-teacher pairs. Firstly, our search space encompasses global features, foreground-background features, instance features, logits response, and localization response as inputs. Then, we construct omni-directional cascaded transformations and obtain the distiller by selecting the advanced distance function and common weight value options. Finally, we present a divide-and-conquer evolutionary algorithm to handle the explosion of the search space. In this strategy, we first evolve the best distiller formulations of individual knowledge inputs and then optimize the combined weights of these multiple distillation losses. DetKDS automates the distillation process without requiring expert design or additional tuning, effectively reducing the teacher-student gap in various scenarios. Based on the analysis of our search results, we provide valuable guidance that contributes to detection distillation designs. Comprehensive experiments on different detectors demonstrate that DetKDS outperforms state-of-the-art methods in detection and instance segmentation tasks. For instance, DetKDS achieves significant gains than baseline detectors: $+3.7$, $+4.1$, $+4.0$, $+3.7$, and $+3.5$ AP on RetinaNet, Faster-RCNN, FCOS, RepPoints, and GFL, respectively. Code at: https://github.com/lliai/DetKDS.

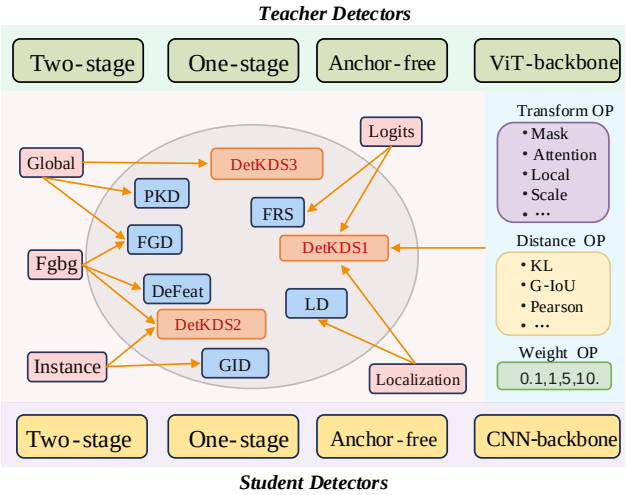

Figure 1. Illustration of DetKDS. Our DetKDS presents the first distiller search framework for different teacher-student pairs. Our search space consists of extensive input knowledge, advanced transformations, distance functions, and loss weights, encompassing almost excellent detection distillation methods (*e.g.*, FGD (Yang et al., 2022a), Defeat (Guo et al., 2021), GID (Dai et al., 2021), LD (Zheng et al., 2022), PKD (Cao et al., 2022)).

1. Introduction

Object detection plays a crucial role in vision tasks (Girshick et al., 2014; Girshick, 2015; Ren et al., 2015; He et al., 2017), including autonomous driving, security monitoring, and pedestrian detection (Sun et al., 2021; Cao et al., 2023b;a). However, the computational complexity and resource requirements of these detectors present challenges for real-time deployment on devices with limited capabilities (Cao et al., 2019). To address this, knowledge distillation (KD) (Hinton et al., 2015) has emerged as a promising way for enhancing smaller student detectors via information from larger, more powerful teacher detectors. This transfer of knowledge allows the student detectors to achieve comparable performance while being more efficient

[1]The Hong Kong University of Science and Technology [2]The Hong Kong University of Science and Technology (Guang Zhou) [3]Institute of Computing Technology, Chinese Academy of Sciences. Correspondence to: Qifeng Liu <liuqifeng@ust.hk>, Yike Guo <yikeguo@ust.hk>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024.

in computation and memory usage.

Knowledge distillation for object detection faces greater challenges than for general tasks (*e.g.*, image classification) due to the added complexities of localization (Zheng et al., 2022). Successful detection requires not only accurate instance classification and localization, but also precise foreground-background context modeling (Guo et al., 2021). To address this, some detection distillation methods focus on localization knowledge and decoupled foreground-background processing. However, designing optimal distillation strategies remains challenging due to several factors. (1) Heterogeneous detectors: detection architectures are diversifying with different backbone networks (*e.g.*, CNNs, ViTs), detection paradigms (*e.g.*, one-stage, two-stage), and labeling schemes (*e.g.*, anchor-based and anchor-free). These different teacher-student detectors perform differently in localization, categorization, and foreground-background learning. While some works attempt to address this problem, they always need to add extra structure or lack generality. (2) Balancing multiple losses: detection distillation involves new losses like instance and localization that expand the loss function space. However, the performance of detection distillation is sensitive to the weight settings of these losses (Zhang & Ma, 2021). Thus, effectively optimizing multiple losses in detection is a critical challenge. (3) Reliance on expert design: selecting appropriate distiller types and debugging hyperparameters for object detection distillation heavily relies on domain expertise. This limits general applicability and scalability to new scenarios and models. These challenges raise two key questions: ***(1) How to find optimal distillation settings for diverse detectors adaptively? (2) Can we minimize expert involvement and efficiently discover optimal KD recipes?***

For the first question, based on empirical investigations, some interesting findings come to light: (1) Some transforms and distance functions benefit distillation gains. For example, channel normalization allows simple MSE loss to work with heterogeneous detectors (Cao et al., 2022). (2) Combining advanced distillation operations can create stronger distillers. Our trials show that cascading attention and normalizing transforms can bring added gains than PKD (Cao et al., 2022) on RetinaNet-R50. (3) Combinations of some common values can balance multiple losses well. These observations inspire us to assemble useful transforms, distance functions, and loss weights and explore their optimization. However, how can we efficiently solve this combination problem? Recent AutoML approaches (*e.g.*, Auto-KD (Li et al., 2023b)) present a promising solution: building a vast search space encompassing options and searching for optimal distillers. However, there are three drawbacks for Auto-KD on detection tasks: (1) Auto-KD only searches on the classification task and transfers to the detection task. Numerous task-customized AutoML

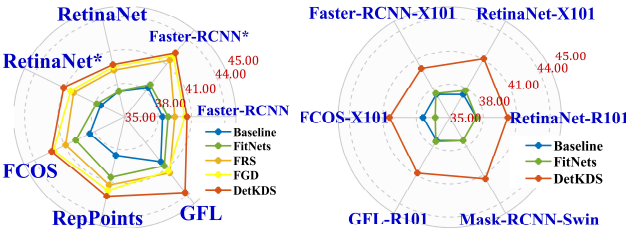

Figure 2. Left: Results of different distilled detectors. *Right:* Results of RetinaNet distilled with different teacher detectors.

methods (Chen et al., 2020) point out that directly searching on the target task is superior to transferring from other tasks. For example, DetNAS (Chen et al., 2019) directly searches the detection backbone and achieves clear gains with similar methods in classification NAS. (2) Auto-KD only searches for general knowledge (*i.e.*, global features and logits) as inputs. Many detection distillation methods argue that distilling on detection-related knowledge (*e.g.*, foreground-background, instance features) is superior to generalized knowledge. For example, GID (Dai et al., 2021) follows relation distillation but works on instance features. (3) Auto-KD's small search space makes it difficult to transfer to complex detectors for satisfactory performance. Meanwhile, Auto-KD only searches small datasets, and its search strategy is not flexible and efficient enough to handle large-scale detection tasks. These limitations demonstrate that detection distillation search is a new research track. and encourage us to develop new task-related search space & strategies to solve them.

Based on the above analysis, we introduce DetKDS, the first automated framework to discover the best distillation settings for different detectors without manual design. DetKDS develops extensive search spaces and efficient search algorithms to automate the process of knowledge distillation in object detection. For the search space, we involve task-customized features and response knowledge as inputs, including global features, foreground-background features, instance features, logits response, and localization response. Then, various options in transform operations, distance functions, and loss weights construct our specific space. For feature inputs, we cascade the attention-based, scale-based, and normalize-based options in different dimensions as the transforms. Advanced distance options like $\ell_{Pearson}$, and ℓ_{SSIM} distance and common weight values are leveraged for feature losses. For response inputs, we opt for some customized options like ℓ_{G-IoU}, and ℓ_{C-IoU} distance in distilling localization information. This extensive search space covers over 30,000 candidates, encompassing existing excellent distillation methods and designs. We divide the task of challenging full distillation configuration search into two sub-tasks, *i.e.*, individual distillation loss search and combined weight optimization for multiple losses. In this

way, we can more efficiently identify the best single distiller and directly transfer it to a different detector to save the search cost. Furthermore, DetKDS incorporates evolutionary algorithms to enable effective exploration of the search space and helps discover potential distillation configurations. To accelerate the search process, DetKDS employs proxy settings and loss-rejection protocol for faster convergence and search efficiency. Finally, we train the student detectors using the discovered distiller without extra manual design and tuning.

Extensive experiments are conducted to elaborate on the effectiveness of DetKDS. On different detectors with varied backbones, our DetKDS achieves state-of-the-art performance among detection KD methods (see Figure 2). For example, using DetKDS, the AP of Faster-RCNN is elevated from 38.4 to 42.5 (+4.1) and RetinaNet from 37.4 to 41.1 (+3.7), benefiting from enhanced knowledge transfer. For anchor-free detectors, DetKDS improves FCOS with a remarkable +4.0 AP boost and RepPoints with a substantial +3.7 AP gain. These results demonstrate the effectiveness of DetKDS across various models. Based on the search results, we derive guidances for KD designs: (1) Global and instance feature distillations outperform other distillations. (2) Channel attention and normalization$_{N,C}$ are the favored transforms. (3) Simple distance functions and loss weights ranging from 0.1 to 10 are sufficient for distillation. To sum up, our paper makes the following contributions:

- We present DetKDS, the first distillation search framework for homogeneous and heterogeneous detector pairs. Our DetKDS allows universal gains of different detectors and frees from expert design and tuning.
- We build the first detection distiller search space, including extensive knowledge inputs, advanced transforms, distance functions, and loss weights. We develop divide-and-conquer evolution and acceleration strategies to efficiently achieve the best solutions.
- Comprehensive experiments verify that DetKDS can achieve state-of-the-art performance on different detectors (*e.g.*, two-stage, one-stage, anchor-free). We also summarize guidances for detection distiller design.

2. Related work

Knowledge Distillation (KD) (Dong et al., 2023a) is extensively uesed for classification tasks using techniques like logits mimicking (Li & Jin, 2022), and feature imitation (Xiaolong et al., 2023). On detection tasks, two-stage detectors (Girshick et al., 2014) use the RPN for proposal generation, while one-stage detectors (Lin et al., 2017) directly predict classification scores and bounding boxes. Anchor-based one-stage detectors (Tian et al., 2019) rely on dense anchor boxes for detection, while anchor-free detectors (Tian et al., 2019) predict center points without predefined anchors. Distilling these detectors needs to address the challenges posed by different types of detectors and region-level location regression & classification. Thus, in contrast to general distillation, detection distillation develops as a new and vital research track. As summarized in Table 1, these methods favor distilling in task-related knowledge (*e.g.*, instance features) and advanced distillation operations (*e.g.*, attention mechanisms). For example, DeFeat (Guo et al., 2021) proposes to decouple the foreground and background features for distillation and FGD (Yang et al., 2022a) tries global relational distillation and tried different distillation operations for foreground-background features. However, manual design and tuning for distillers on large-scale and complex detection tasks are challenging. Compared to these handcrafted methods (Cao et al., 2023c), our DetKDS focus on searching with full distillation configurations for different detectors. In contrast to Auto-KD methods (Li et al., 2023b;a) on classification tasks, our DetKD searches directly on detection tasks and involves different search spaces with task-customized knowledges & operations (see Table 2 & 3) and efficient search strategies (see Table 4).

Table 1. Summary of recent SOTA detection distillation methods.

Method	KD op types	KD loss types	Algorithm note
GID (2021)	relation-based	$\mathcal{L}_{ins}$+$\mathcal{L}_{logits}$+$\mathcal{L}_{loc}$	instance relation distillation
FRS (2021)	attention-based	$\mathcal{L}_{global}$+$\mathcal{L}_{logits}$+$\mathcal{L}_{loc}$	feature-richness score to select key features
FKD (2021)	attention-based	$\mathcal{L}_{global}$+$\mathcal{L}_{fbgb}$	attention-guided and non-local distillation
Defeat (2021)	decoupled-based	$\mathcal{L}_{global}$+$\mathcal{L}_{fbgb}$+$\mathcal{L}_{logits}$+$\mathcal{L}_{loc}$	decoupled features
PKD (2022)	normalize-based	$\mathcal{L}_{global}$	imitate features via pearson correlation
LD (2022)	location-based	$\mathcal{L}_{global}$+$\mathcal{L}_{loc}$	localization representation of bounding box
FGD (2022a)	attention-based	$\mathcal{L}_{global}$+$\mathcal{L}_{fbgb}$	separates fbgb, global relation distillation
SKD (2022)	SSIM	$\mathcal{L}_{global}$	structural similarity for feature importance

3. Methodology

3.1. Detection Distiller Search Space

Unified DetKD optimization objectives. Detection distillation aims to minimize disparities between features f_T & responses p_T of teacher and features f_S & responses p_S of student. As shown in Figure 3, DetKDS unifies most detection distillation methods into one search space, with the total loss $\mathcal{L}$ as follows:

$$\mathcal{L} = \mathcal{L}_{orig} + \mathcal{L}_{global} + \mathcal{L}_{fgbg} + \mathcal{L}_{ins} + \mathcal{L}_{logits} + \mathcal{L}_{loc}, \quad (1)$$

where $\mathcal{L}_{orig}$ is the original task loss for detectors. $\mathcal{L}_{global}$, $\mathcal{L}_{fgbg}$, and $\mathcal{L}_{ins}$ are feature KD losses on global features, foreground-background features, and instance features. $\mathcal{L}_{logits}$ and $\mathcal{L}_{loc}$ are logits and localized distillation losses on the prediction heads. We extract the global hierarchical feature of teacher-students for $\mathcal{L}_{global}$, which can be easily applied to different detectors. For $\mathcal{L}_{fgbg}$, we set a binary mask to separate foreground the background features generated by the ground-truth boxes. Similarly, we use the ground-truth boxes as cropping to obtain instance features.

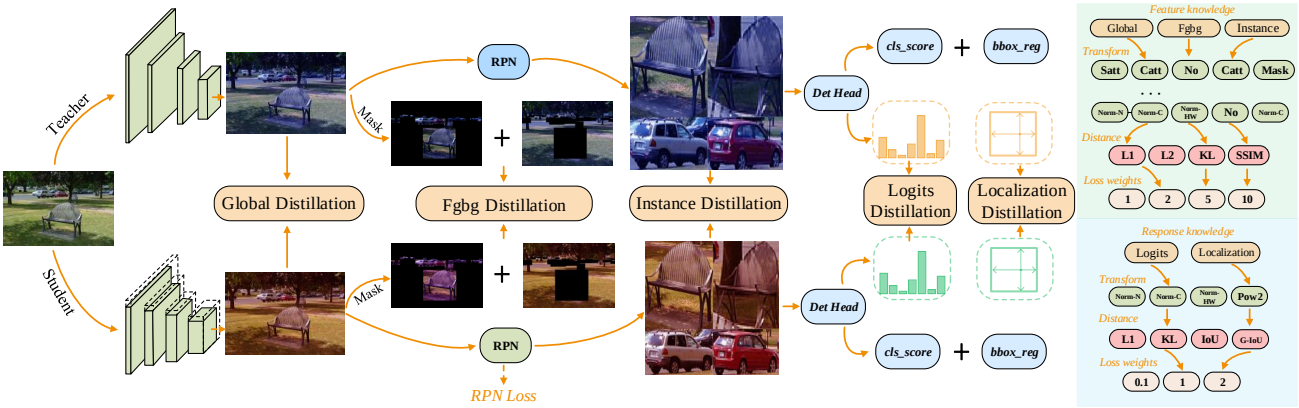

Figure 3. Overview of the detection distillation search framework of our DetKDS. We search the best transform, distance, and loss weight options to create comprehensive task-customized distillation strategies, including distillation of global feature, foreground-background (fgbg) feature, instance feature, logits responses, and localization responses between teacher-student detectors.

Their detailed formulation is defined as follows:

$$\mathcal{L}_{\{global,fgbg,ins\}} = \mathcal{W}_f \times \mathcal{D}_f\big(\mathcal{T}_{f1}\langle\mathcal{T}_{f2}\langle\mathcal{T}_{f3}\langle f_S, f_T\rangle\rangle\rangle\big), \tag{2}$$

where $\mathcal{W}_f$ is the loss weighted factor, and $\mathcal{T}_f$ is feature transformations. $\langle\cdot,\cdot\rangle$ means to perform the same operation on two primitives. We assign three transform options as $\mathcal{T}_{f1}\rightarrow \mathcal{T}_{f2} \rightarrow \mathcal{T}_{f3}$ for the transformation part to ensure effective processing of the feature knowledge. $\mathcal{D}_f(\cdot,\cdot)$ as distance function measuring the difference of feature representations. For response knowledge, we use the classification and localization outputs for response distillation, as:

$$\mathcal{L}_{\{logits,loc\}} = \mathcal{W}_p \times \mathcal{D}_p\big(\mathcal{T}_p\langle p_S, p_T\rangle\big), \tag{3}$$

where $\mathcal{W}_p$ is loss weight, $\mathcal{T}_p$ is response transformations, and $\mathcal{D}_p(\cdot,\cdot)$ is distance function measuring the difference of responses. Note that we follow the same LD implementation (Zheng et al., 2022) to capture the localization outputs.

Search space options. Our search space includes various advanced options for the transformation and distance functions, the common values of loss weights. Specifically, for feature knowledge, different detectors tend to have large distributional differences. Therefore, we assign three cascade transform options as *attention-transform→scale-transform→normalize-transform* for the transformation part to ensure effective processing of the input knowledge. As shown in Table 2, we opt to perform attention transformations for feature $f_{N,C,H,W}$ in the channel C and spatial HW dimensions, as:

$$\mathcal{T}_{catt} = HW \times \psi(f_C) \times f, \mathcal{T}_{satt} = C \times \psi(f_{HW}) \times f, \tag{4}$$

where ψ stands for softmax function. In addition, local and multi-scale features are important for detection tasks (Li et al., 2020; Pengguang et al., 2021), so we consider local and dimensional transformations for different dimensions:

$$\mathcal{T}_{scale-n} = f^{N,C,H/n,W/n}, \mathcal{T}_{local-n} = f^{n^2\times N,C,H/n,W/n}, \tag{5}$$

where the scale transform is implemented via pooling, and the local transform treats various local region features as different samples. Finally, we consider the normalization operation on $\{N, C, HW\}$ dimensions:

$$\mathcal{T}_{norm_{\{N,C,HW\}}} = (f_{\{N,C,HW\}} - \mu_{\{N,C,HW\}})/\sigma_{\{N,C,HW\}}, \tag{6}$$

where μ and σ represent the mean and standard deviation. In addition, we consider some desirable distance functions with common loss weights as options (see Table 2). For response knowledge, we employ a single transform from different activation and normalization options. Considering differences between logits and localization response, we individually configure the logits distance options (*e.g.*, $\ell_{Pearson}$) and

Table 2. Options for detection distillers in DetKDS.

	Feature knowledge
Transform	attention-based: $satt, catt, mask$ scale-based: $scale_{1,2,4}, local_{1,2,4}$ normalize-based : $norm_{HW,C,N,Min-max}$
Distance	$\ell_{smooth1}, \ell_1, \ell_2, \ell_{KL}, \ell_{Cosine}, \ell_{Pearson}, \ell_{SSIM}$
Weight	$0.1, 0.5, 1, 2, 5, 8, 10$
	Logits response
Transform	$norm_{N,C,Min-max}, reshape_{N,C}$
Distance	$\ell_{Smooth\_l1}, \ell_1, \ell_2, \ell_{KL}, \ell_{Cosine}, \ell_{Pearson}, \ell_{Cor}$
Weight	$0.1, 0.5, 1, 2, 5, 8, 10$
	Localization response
Transform	$softmax_{N,C}, pow2, sigmoid$
Distance	$\ell_{IoU}, \ell_{C-IoU}, \ell_{E-IoU}, \ell_{G-IoU}, \ell_{D-IoU}$
Weight	$0.1, 0.5, 1, 2, 5, 8, 10$

Algorithm 1 Divide-and-conquer Evolution in DetKDS

Input: Search space $\mathcal{S}$, population $\mathcal{P}$, max iteration $\mathcal{T}$, sample ratio r, sampled pool $\mathcal{R}$, topk k.
Output: Distiller with highest validation AP score.

for $L = L_{global}, L_{fgbg}, L_{ins}, L_{logits}, L_{loc}$ **do**
  $\mathcal{P}0$:= Initialize population(P_i); Sample pool $\mathcal{R} := \emptyset$;
  for $i = 1, 2, \ldots, \mathcal{T}$ **do**
    Clear sample pool $\mathcal{R} := \emptyset$;
    Randomly select $\mathcal{R} \in \mathcal{P}$;
    Parent G_i^p := RandomSelect($G_{i\,k}$) where $G_i k$:= GetTopk($\mathcal{R}$, k);
    Mutate G_i^m := MUTATE(G_i^p);
    Randomly Sample E;
    Compare E with G_i^m and append better distiller to $\mathcal{P}$;
    Remove the distiller with lowest validation AP score.
  end for
end for
$\mathcal{P}1$:= Random sample population($\mathcal{N}$, $\mathcal{S}$);
for Candidates$\{\mathcal{W}_i\} \in \mathcal{W}$ **do**
  Get $\mathcal{W}_1 L_{global} + \mathcal{W}_2 L_{fgbg} + \mathcal{W}_4 L_{ins} + \mathcal{W}_4 L_{logits} + \mathcal{W}_5 L_{loc}$;
  Get validation AP score ;
  Crossover and mutation $\mathcal{W}_i$.
end for

localization distance options (*e.g.*, ℓ_{G-IoU} and ℓ_{D-IoU}). ℓ_{IoU}, ℓ_{G-IoU} and ℓ_{D-IoU} differ in optimizing the localization response of the teacher-student detectors.

Novelty of our search space. (1) Comprehensive. DetKDS not only considers common distillations in global features and logits responses but also involves task-specific distillations (*e.g.*, foreground-background and localization) in detection tasks. As shown in Table 3, DetKDS contains many task-related KD designs including various transforms and distances. **(2) Innovative:** Our DetKDS not only includes many existing detection distillation methods but also extends the advanced operations to omni-dimensions and new combinations. For example, we scale the normalize op to $\{N, C, HW\}$ dimensions, and most of our combined advanced transformations and distance functions are different from previous forms. Please see Appendix Sec. D for more details of our search space.

3.2. Divide and Conquer Evolution Search

Divide-and-Conquer. Different from the naive strategy of Auto-KD that takes distiller search as a whole problem, we develop a novel divide-and-conquer strategy in Alg. 1 by making use of the modular nature of the search space. We provide analyses to confirm that our divide-and-conquer strategy offers theoretical advantages regarding effective search space exploration, convergence guarantees, and computational complexity:

The overall distillation search space is modeled as a high-dimensional discrete space Ω, where each point $\omega \in \Omega$ represents a specific policy configuration. The goal is to find the optimal policy ω^* that minimizes the loss function $L(\omega)$:

$$\omega^* = \arg\min_{\omega \in \Omega} L(\omega) \tag{7}$$

DetKDS employs a divide-and-conquer strategy by breaking Ω into subspaces $\Omega = \Omega_1 \times \Omega_2 \times ... \times \Omega_m$, where each Ω_i corresponds to the search space for an individual knowledge component (e.g. global features, instance features). Theoretically, this decomposition can improve convergence rates. If the search on each Ω_i is modeled as an absorbing Markov chain with stationary distribution π_i and hitting time H_i to ω^i, then the overall hitting time H to ω satisfies:

$$\mathbb{E}[H] \leq \sum_i \mathbb{E}[H_i] \tag{8}$$

Assuming each $|\Omega_i| = n$ and $|\Omega| = n^m$, this divide-and-conquer reduces the complexity from $\mathcal{O}(n^m)$ to $\mathcal{O}(m \cdot n)$. Furthermore, the decomposition can provide better approximation guarantees by avoiding high dimensionality issues. If each $L(\omega_i)$ has approximation ratio α_i, the overall $L(\omega^*)$ is within a factor $\prod_i \alpha_i$ of the global optimal, which is tighter than evolving the full Ω directly. These analyses of the convergence rates clearly demonstrate how our DetKDS effectively explores the vast search space and facilitates the transfer of knowledge between different detectors. These theoretical guarantees complement the empirical results and provide a deeper understanding of the optimization advantages offered by our search methodology.

Evolution production and acceleration. Firstly, we search in parallel for the optimal distillation configuration for the single input. During the search, our algorithm evolves the population over generations using genetic operators such as selection, crossover, and mutation to generate better validation of AP results as fitting objectives. After obtaining the best individual KD functions, we additionally search for their combined weights to different detectors. In addition, we employ several strategies to accelerate the search space: **(1) Distiller filter.** We will filter out candidates with excessive loss values ($>$100 & nan) during search. **(2) Proxy settings.** We only adopt 20% subsets for searching and early stop the validation process once the student model performs well enough to determine the quality of the candidates. This strategy improves search efficiency and guarantees strong correlation and true performance (see Appendix Sec. A.1).

Advantages of our search algorithm can be summarized as: (1) Effective. Our search space includes more than 30,000 candidates and many weak solutions. Our divide-and-conquer manner effectively shrinks the search space and allows for easy procedure refinement. For example, we can remove some weak losses that are difficult to optimize from our combined search. **(2) General.** Because we divide the search object into sub-parts, we can avoid some repeating searches by transferring some excellent losses for different detectors. As shown in Table 5, we observe that the $\mathcal{L}_{\text{global}}$ (catt $\rightarrow$ scale$_2$ $\rightarrow$ norm$_N$) can generalize well to different student pairs. **(3) Efficient.** During the search process, our bag of acceleration strategies achieves at least $100\times$ speed-ups in the process.

Comparison with Auto-KD. Our DetKDS differs from

Table 3. Comparison of search space for DetKDS with Auto-KD.

Methods	Space size	Knowledge	Typical $\mathcal{T}$ ops	Typical ℓ ops	Included KDs
Auto-KD	2,000	$\mathcal{L}_{global}$,$\mathcal{L}_{logits}$	mask	ℓ_2, ℓ_{Gram}	SP, ICKD, CWD
DetKDS	30,000	$\mathcal{L}_{global}$,$\mathcal{L}_{fgbg}$,$\mathcal{L}_{ins}$ $\mathcal{L}_{ins}$,$\mathcal{L}_{logits}$	$norm_{HW,C,N}$ $softmax_{N,C}$	$\ell_{Pearson}$, ℓ_{SSIM} ℓ_{G-IoU}, ℓ_{D-IoU}	FGD, PKD, GID, LD Defeat, FKD, PGD

Table 4. Comparison of search alg. for DetKDS with Auto-KD.

Methods	Search task	Det-distiller	Search alg.	Typical strategy
Auto-KD	classification	transfered	MTCS	None
DetKDS	detection	auto-search	Evolution	divide-and-conquer

Auto-KD (Li et al., 2023b) in: **(1) Search space.** As shown in Table 4, DetKDS constructs a tailored search space, including inputs like foreground-background features, instance features, localization responses, etc., and specialized operations like local/multi-scale ops and advanced distance metrics like IoU-based losses - all tailored specifically for localization and detection tasks. As a result, our search space covers more existing detection distillation methods than Auto-KD. **(2) Search tasks.** As shown in Table 4, our method directly searches on the detection task, while Auto-KD only searches for classification. DetKDS Handling diverse detectors and architectural variations than Auto-KD. Object detection datasets/models are typically more complex and larger-scale compared to Auto-KD. DetKDS employs proxy settings and evolutionary search strategies to efficiently navigate this large search space of complex object detectors. **(3) Search algorithms.** DetKDS uses a novel divide-and-conquer strategy to first search individual distillation losses before combining them - a technique motivated by the unique challenges of balancing multiple losses (global, instance, localization, etc.) in detection distillation, which is not in Auto-KD. For search engines, we employ more flexible and efficient evolutionary search compared to Monte Carlo tree search of Auto-KD.

3.3. Search Result Analysis

Table 5 presents searched distillers for different detectors. In addition, we provide stripping experiments for individually searched losses in Table 11 and extensive experiments on homogeneous and heterogeneous student-teacher pairs in Table 7, 9 and 8. By analyzing these extensive results, we can summarize some practical guidance for distillers design and tuning on object detection.

For various types of knowledge: In our experiments of Table 11, global distillation and instance distillation enjoy superior performance than other losses. Response distillations are weaker than feature distillations on detection tasks, consistent with other work's findings (Huang et al., 2022).

For different distillation options: We can derive several interesting common patterns from the specific forms in Table 5. For feature transformations, channel attention, scale operations, and normalization in $\{N, C\}$ dimensions are commonly employed options. Similarly, sigmoid and batch-wise normalization are always adopted for the response transformation. For the distance function, we observe that some simple options (*e.g.*, ℓ_{l1}) can be generalized to different detectors. This could be attributed to that our transformation part already deals with knowledge well, and these easily optimizable distance functions are always selected in our frameworks. In addition, ℓ_{C-IoU} and ℓ_{E-IoU} are more suitable for localization distillation. For the loss weights, we can pick the optimal solution from a range of 0.1 to 10.

Table 5. Detailed forms of found distillers of DetKDS.

Input	Transform $\mathcal{T}$	Distance ℓ
	Two-stage detectors	
$\mathcal{L}_{global}$	catt $\rightarrow$ scale$_2$ $\rightarrow$ norm$_N$	ℓ_{l1}
$\mathcal{L}_{fgbg}$	catt $\rightarrow$ norm$_C$	$\ell_{\text{Smooth\_l1}}$
$\mathcal{L}_{ins}$	mask $\rightarrow$ scale$_1$ $\rightarrow$ norm$_C$	ℓ
$\mathcal{L}_{logits}$	softmax$_C$	ℓ_{Pearson}
$\mathcal{L}_{loc}$	sigmoid	$\ell_{\text{C-IoU}}$
	One-stage detectors	
$\mathcal{L}_{global}$	catt $\rightarrow$ scale$_2$ $\rightarrow$ norm$_N$	ℓ_{l1}
$\mathcal{L}_{fgbg}$	local$_4$ $\rightarrow$ norm$_{HW}$	ℓ_{l1}
$\mathcal{L}_{ins}$	catt $\rightarrow$ scale$_1$ $\rightarrow$ norm$_C$	$\ell_{\text{Smooth\_l1}}$
$\mathcal{L}_{logits}$	norm$_{batch}$	ℓ_{Pearson}
$\mathcal{L}_{loc}$	sigmoid	$\ell_{\text{C-IoU}}$
	Anchor-free detectors	
$\mathcal{L}_{global}$	catt $\rightarrow$ scale$_2$ $\rightarrow$ norm$_N$	ℓ_{l1}
$\mathcal{L}_{fgbg}$	mask $\rightarrow$ scale$_1$ $\rightarrow$ norm$_C$	ℓ_{l1}
$\mathcal{L}_{ins}$	satt $\rightarrow$ scale$_4$ $\rightarrow$ norm$_N$	ℓ_{l1}
$\mathcal{L}_{logits}$	norm$_{batch}$	ℓ_{l2}
$\mathcal{L}_{loc}$	sigmoid	$\ell_{\text{E-IoU}}$

Generalization of distillers: If the searched distillers can be effectively applied to different detectors, it becomes highly meaningful as it eliminates the need for additional search efforts. We find that our searched distillers for $\mathcal{L}_{global}$, $\mathcal{L}_{logits}$, and $\mathcal{L}_{loc}$ have the same or similar forms for different detectors. In contrast, the search results of $\mathcal{L}_{fgbg}$ and $\mathcal{L}_{ins}$ are dissimilar, which may be related to the properties of the detectors.

Take-home messages: In summary, we can create distillers for new detectors using feature knowledge as inputs. These functions involve transformations like channel attention, scaling, and normalization$_{N,C}$. We can use simple distance functions and choose loss weights between $0.1 \sim 10$.

4. Experiments

In this section, we first perform extensive experiments to evaluate the efficiency of our DetKDS in distilling homo-

Table 6. **Comparison with state-of-the-art object detection KD methods on COCO val set.** Results of other methods are referenced from their original results. DetKD (feature) refers to DetKD with $\mathcal{L}_{\{global,fgbg,ins\}}$. DetKDS report best results of DetKD with $\mathcal{L}_{\{global,fgbg,ins\}}$ or $\mathcal{L}_{\{global,fgbg,ins,logits,loc\}}$.

Method	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
Two-stage detectors						
Faster RCNN-R101 (T)	39.8	60.1	43.3	22.5	43.6	52.8
Faster RCNN-R50 (S)	38.4	59.0	42.0	21.5	42.1	50.3
FitNets (2014)	38.9 (0.5↑)	59.5	42.4	21.9	42.2	51.6
GID (2021)	40.2 (1.8↑)	60.7	43.8	22.7	44.0	53.2
FRS (2021)	39.5 (1.1↑)	60.1	43.3	22.3	43.6	51.7
FGD (2022a)	40.4 (2.0↑)	-	-	22.8	44.5	53.5
PKD (2022)	40.5 (2.1↑)	60.9	44.4	22.6	44.8	53.1
Auto-KD (2023b)	40.1 (1.7↑)	60.6	43.7	22.7	44.0	52.8
DiffKD (2023b)	40.6 (2.2↑)	60.9	43.9	23.0	44.5	54
DetKDS	41.0 (2.6↑)	61.6	45.0	24.5	45.6	53.7
DetKDS (feature)	40.6 (2.2↑)	60.8	44.1	22.9	44.8	53.5
One-stage detectors						
RetinaNet-R101 (T)	38.9	58.0	41.5	21.0	42.8	52.4
RetinaNet-R50 (S)	37.4	56.7	39.6	20.0	40.7	49.7
FitNets	37.4 (0.0↑)	57.1	40.0	20.8	40.8	50.9
GID (2021)	39.1 (1.7↑)	59.0	42.3	22.8	43.1	52.3
FRS (2021)	39.3 (1.9↑)	58.8	42.0	21.5	43.3	52.6
FGD (2022a)	39.6 (2.2↑)	-	-	22.9	43.7	53.6
Auto-KD (2023b)	39.5 (2.1↑)	58.5	42.4	22.8	43.5	53.3
DiffKD (2023b)	39.7 (2.3↑)	58.6	42.1	21.6	43.8	53.3
MasKD (2023a)	39.8 (2.4↑)	59.0	42.5	21.5	43.9	54.0
DetKDS	40.2 (2.8↑)	59.5	43.0	22.6	44.3	53.4
DetKDS (feature)	39.8 (2.4↑)	58.7	42.8	21.8	43.8	53.5
One-stage detectors						
GFL-R101 (T)	44.9	63.1	49.0	28.0	49.1	57.2
GFL-R50 (S)	40.2	58.4	43.3	23.3	44.0	52.2
FitNets (2014)	40.7 (0.5↑)	58.6	44.0	23.7	44.4	53.2
Defeat (2021)	40.8 (0.6↑)	58.6	44.2	24.3	44.6	53.7
FGFI (2019)	41.1 (0.9↑)	58.8	44.8	23.3	45.4	53.1
FGD (2022a)	41.3 (1.1↑)	58.8	44.8	24.5	45.6	53.0
GID (2021)	41.5 (1.3↑)	59.6	45.2	24.3	45.7	53.6
SKD (2022)	42.3 (2.1↑)	60.2	45.9	24.4	46.7	55.6
LD (2022)	43.0 (2.8↑)	61.6	46.6	25.5	47.0	55.8
PKD (2022)	43.3 (3.1↑)	61.3	46.9	25.2	47.9	56.2
DetKDS	43.7 (3.5↑)	62.1	47.4	26.9	48.0	56.2
DetKDS (feature)	43.7 (3.5↑)	62.1	47.4	26.9	48.0	56.2
Anchor-free detectors						
FCOS-R101 (T)	40.8	60.0	44.0	24.2	44.3	52.4
FCOS-R50 (S)	38.5	57.7	41.0	21.9	42.8	48.6
FitNets (2014)	39.9 (1.4↑)	58.6	43.1	23.1	43.4	52.2
GID (2021)	42.0 (3.5↑)	60.4	45.5	25.6	45.8	54.2
FRS (2021)	40.9 (2.4↑)	60.3	43.6	25.7	45.2	51.2
FGD (2022a)	42.1 (3.6↑)	-	-	27.0	46.0	54.6
Auto-KD (2023b)	42.0 (3.5↑)	60.4	45.5	25.9	45.8	54.4
DiffKD (2023b)	42.4 (3.9↑)	61.0	45.8	26.6	45.9	54.8
DetKDS	42.5 (4.0↑)	60.9	46.1	25.7	46.7	54.1
DetKDS (feature)	42.3 (3.8↑)	60.8	45.6	26.1	46.2	54.7

Table 7. **Comparison with state-of-the-art object detection KD methods on DETR detectors.**.

Method	AP	AP_S	AP_M	AP_L
Deformable DETR-R101 (T)	45.5	27.5	48.7	60.3
Deformable DETR-R50 (S)	44.1	27.0	47.4	58.3
FGD (2022a)	44.1 (+0.0↑)	25.9	47.7	58.8
MGD (2022b)	44.0 (-0.1)	25.9	47.3	58.6
DetKDS	46.2 (+2.1↑)	29.1	49.7	60.5

Table 8. **Results with heterogeneous teachers on COCO val set.** Note that all students are distilled via DetKDS.

Detector	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
RetinaNet-R50 (S)	37.4	56.7	39.6	20.0	40.7	49.7
Faster-RCNN-X101 (T)	41.2	61.7	44.8	23.9	44.9	54.3
DetKDS	40.1 (2.7↑)	59.3	42.9	22.3	43.0	53.3
FCOS-X101 (T)	42.7	62.5	45.7	26.0	46.5	54.7
DetKDS	40.4 (3.0↑)	59.8	43.0	22.8	44.3	53.8
GFL-R101 (T)	44.9	63.1	49.0	28.0	49.1	57.2
DetKDS	40.8 (3.4↑)	59.9	43.7	24.4	45.2	54.4
Mask-RCNN-Swin-S (T)	48.2	69.8	52.9	32.1	51.8	62.7
DetKDS	41.4 (4.0↑)	60.8	44.4	23.4	45.1	55.5

geneous and heterogeneous detectors. Then, we extend DetKDS for instance segmentation and provide detailed analyses of our individual losses, search algorithms, and generalizations. Please see Appendix Sec. B for more visualizations.

4.1. Datasets and Implement Details

We evaluate our method on the COCO dataset (Lin et al., 2014), which contains 80 object classes. For search settings, we utilize all searches on the subsets of COCO training set (*i.e.*, mini-COCO), which consists of 25K training images and 5K validation images. **Note that our validation set does not overlap with the test set, and we search using the validation results to ensure fair comparisons.** We configure 20 iterations of parallel searching for individual losses and 40 iterations for combined weights for multiple losses. We set training to one epoch for each search iteration. For EA settings, we set ($\mathcal{P}$, $\mathcal{T}$, r, k) in Alg. 1 as $(20, 40, 0.9, 5)$. After searching, we train student detectors with the best distiller on the full COCO dataset, including 120K train images. Following the same training settings as FGD, we develop our experiments using 8 NVIDIA V100 GPUs with a mini-batch of two images per GPU. We train all the detectors for 24 epochs with SGD optimizer, in which the momentum is 0.9, and the weight decay is 0.0001.

4.2. Main Results on Homogeneous Detectors

We first compare the different distillation methods under various homogeneous detectors with ResNet101 as the teacher backbone and ResNet50 as the student backbone in Table 7. The results indicate that our DetKDS surpasses existing

Table 9. **Results with heterogeneous backbone on COCO val set.** *CM RCNN*: Cascade Mask RCNN. DetKDS report best results of DetKD with $\mathcal{L}_{\{global,fgbg,ins\}}$ or $\mathcal{L}_{\{global,fgbg,ins,logits,loc\}}$.

Method	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
Two-stage detectors						
CM RCNN-X101 (T)	45.6	64.1	49.7	26.2	49.6	60.0
Faster RCNN-R50 (S)	38.4	59.0	42.0	21.5	42.1	50.3
FGFI (2019)	39.1 (0.7↑)	59.8	42.8	22.2	42.9	51.1
COFD (2019)	38.9 (0.5↑)	60.1	42.6	21.8	42.7	50.7
FKD (2021)	41.5 (3.1↑)	62.2	45.1	23.5	45.0	55.3
CWD (2021)	41.7 (3.3↑)	62.0	45.5	23.3	45.5	55.5
KD-Zero (2023a)	41.9 (3.5↑)	62.7	45.5	23.6	45.6	55.6
FGD (2022a)	42.0 (3.6↑)	-	-	23.7	46.4	55.5
MGD (2022b)	42.1 (3.7↑)	-	-	23.7	46.7	56.1
DiffKD (2023b)	42.2 (3.8↑)	62.8	46.0	24.2	46.6	55.3
MasKD (2023a)	42.4 (4.0↑)	62.9	46.8	24.2	46.7	55.9
DetKDS	42.5 (4.1↑)	62.9	46.2	24.4	46.6	56.2
DetKDS (feature)	42.3 (3.9↑)	62.8	46.2	24.8	46.1	56.0
One-stage detectors						
RetinaNet-X101 (T)	41.6	61.4	44.3	23.9	45.5	54.5
RetinaNet-R50 (S)	37.4	56.7	39.6	20.0	40.7	49.7
COFD (2019)	37.8 (0.4↑)	58.3	41.1	21.6	41.2	48.3
FKD (2021)	39.6 (2.2↑)	58.8	42.1	22.7	43.3	52.5
FRS (2021)	40.1 (2.7↑)	59.5	42.5	21.9	43.7	54.3
CWD (2021)	40.8 (3.4↑)	60.4	43.4	22.7	44.5	55.3
FGD (2022a)	40.4 (3.0↑)	-	-	23.4	44.7	54.1
PKD (2022)	40.8 (3.4↑)	60.3	43.4	23	45.1	54.7
MasKD (2023a)	40.9 (3.5↑)	60.1	43.6	22.8	45.3	55.1
DiffKD (2023b)	40.7 (3.3↑)	60.0	43.2	22.2	45.0	55.2
KD-Zero (2023a)	40.9 (3.5↑)	60.4	43.5	23.2	45.2	54.8
DetKDS	41.1 (3.7↑)	60.4	43.9	23.2	45.1	55.1
DetKDS (feature)	41.0 (3.8↑)	60.2	43.6	23.0	45.2	55.0
Anchor-free detectors						
RepPoints-X101 (T)	44.2	65.5	47.8	26.2	48.4	58.5
RepPoints-R50 (S)	38.6	59.6	41.6	22.5	42.2	50.4
FKD (2021)	40.6 (2.0↑)	61.7	43.8	23.4	44.6	53.0
CWD (2021)	42.0 (3.4↑)	63.0	45.3	24.1	46.1	55.0
FGD (2022a)	41.3 (2.7↑)	-	-	24.5	45.2	54.0
MasKD (2023a)	41.8 (3.2↑)	62.6	45.1	24.2	45.4	55.2
DiffKD (2023b)	41.7 (3.1↑)	62.6	44.9	23.6	45.4	55.9
MKD	42.2 (3.6↑)	63.0	45.6	24.3	46.4	55.7
DetKDS	42.3 (3.7↑)	63.1	45.8	24.1	46.4	55.9
DetKDS (feature)	42.3 (3.7↑)	63.1	45.8	24.1	46.4	55.9

Table 10. Results of instance segmentation via Cascade Mask R-CNN (ResNeXt-101) as teacher. AP means Mask AP.

Method	AP	AP_S	AP_M	AP_L
Mask RCNN-R50 (S)	35.4	19.1	38.6	48.4
FKD (2021)	37.4	19.7	40.5	52.1
FGD (2022a)	37.8	17.1	40.7	56.0
MGD (2022b)	38.1	17.1	41.1	56.3
DetKDS	38.4	18.8	41.3	55.8

Table 11. Ablation study of individual distillation loss with RetinaNet-R50 as student, RetinaNet-X101 as teacher.

Method	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
Baseline	37.4	56.7	39.6	20.0	40.7	49.7
$\mathcal{L}_{global}$	41.0	60.2	43.6	23.0	45.2	55.0
$\mathcal{L}_{fgbg}$	40.7	60.2	43.4	23.8	44.6	53.9
$\mathcal{L}_{ins}$	40.9	60.7	43.4	23.5	44.9	53.9
$\mathcal{L}_{loc}$	39.4	58.3	42.3	22.6	43.5	51.2
$\mathcal{L}_{logits}$	38.6	57.9	41.0	21.6	42.0	51.8

Table 12. Ablation study of search algorithm on RetinaNet-R50 (student) and RetinaNet-X101 (teacher). *DC*: Divide-and-Conquer, *DF*: Distiller Filter, *MCTS*: Monte-Carlo tree search.

DC	DF	Search Alg.	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
✓	✓	Evolution	41.1	60.4	43.4	23.8	44.6	53.4
✓	✓	MCTS	40.8	60.1	43.2	23.2	44.3	53.1
✓	✓	Random	40.5	59.7	43.0	22.6	44.3	53.2
-	✓	Evolution	40.4	59.6	43.3	23.7	44.9	54.2
✓	-	Evolution	39.8	58.6	42.3	22.3	43.7	54.4

state-of-the-art methods by meaningful margins across popular detectors. For example, the student detectors of Faster-RCNN, RetinaNet, GFL, and FCOS with DetKDS exhibit significant $+2.6$, $+2.8$, $+3.5$, and $+4.0$ AP improvements on the COCO dataset, respectively. Compared to other methods, DetKDS obtains improvements of 0.5 AP over Auto-KD, and consistently outperforms GID across detectors, with larger AP gains of 2.6 vs. 1.8 for Faster-RCNN and 4.0 vs. 3.5 for FCOS. With vanilla feature knowledge, our DetKDS (feature) also obtains more stable gains than FGD. As shown in Table 7, distilling Deformable DETR consistently achieves an AP of 46.2 in our method. These state-of-the-art improvements over other methods highlight the effectiveness of DetKDS at compressing detection models.

4.3. More Results with Heterogeneous Detectors

In contrast to current techniques only for homogeneous detectors, DetKDS can also be well-generalized to heterogeneous detector pairs. To verify this, we individually conduct experiments under various detectors with heterogeneous backbones and cross-detector paradigms. In Table 9, we first configure the heterogeneous backbones like ResNeXt-101 and ResNet50 for teacher-student detectors. DetKDS achieves $3.77 \sim 4.1$ gains on student baseline and $0.37 \sim 3.6$ higher AP gains over other methods on all detector paradigms. Particularly for the anchor-free detector, DetKDS improves the student by 1 AP compared to FGD. DetKDS achieves significant performance gains, surpassing prior work on homogeneous networks, and establishes itself as the state-of-the-art in distilling object detection mod-

Table 13. Search time (hours) of search iterations on 8× V100 GPUs for RetinaNet-R50 with RetinaNet-X101 as teacher.

Iters.	Time	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
20	2.5 h	40.8	60.2	43.6	22.8	44.9	55.5
30	3.7 h	40.9	60.3	43.8	22.9	45.1	55.0
40	5 h	41.1	60.4	43.4	23.8	44.6	53.4

Table 14. Results of DetKDS for lightweight detectors With ResNet-X101-based RetinaNet as teacher.

Student	**DetKDS**	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
ResNet-18		32.6	50.6	34.6	17.8	35.2	43.5
	✓	36.1	54.3	38.0	18	39.5	50.5
ResNet-34		34.9	54.8	37.0	20.9	38.9	44.8
	✓	39.8	58.6	42.3	22.3	43.7	54.4
ResNet-50		37.4	56.7	39.6	20.0	40.7	49.7
	✓	41.1	60.4	43.4	23.8	44.6	53.4

els with different structures. Then, we apply Auto-DeKD for distilling RetinaNet with cross-paradigm teachers. As shown in Table 8, our DetKDS demonstrates remarkable $2.7 \sim 4.0$ AP gains in AP across different teachers. These observations suggest that DetKDS unlocks newfound potential for heterogeneous scenarios where detectors have different backbones, paradigms and label assignments.

4.4. Extended Instance Segmentation Experiments

For instance segmentation, we conduct experiments on Mask RCNN (He et al., 2017). As shown in Table 10, our DetKDS obtains 3.0 Mask AP gains for Mask RCNN and surpasses the other methods on instance segmentation.

4.5. Ablation Studies

Analysis of different losses. We perform ablation studies for individual loss on RetinaNet. As demonstrated in Table 11, each type of loss yields different improvements. The $\mathcal{L}_{global}$ loss outperforms the others, followed by the $\mathcal{L}_{ins}$ and $\mathcal{L}_{fgbg}$. The $\mathcal{L}_{logit}$ and $\mathcal{L}_{loc}$ losses also obtain improvements over the baseline. Thus, for different detector distillations, we favor the feature KD losses and select other KD losses during the combined multi-loss search.

Analysis of search algorithm. (1) As shown in Table 12 and Figure 4, our Evolution Algorithm (EA) significantly outperforms random search and obtains some gains than MCTS in Auto-KD. These results confirm that EA can flexibly control the exploration process. In addition, our divide-and-conquer and distiller filter strategies play important roles in pruning search space, making the search process more efficient and effective. (2) As shown in Figure 4, our search algorithm with different initialization seeds performs searching stably.

Analysis of search efficiency. We dramatically optimize the search cost by employing distiller filter, proxy settings, and transferring found distiller. As shown in Table 13, our default settings need **only** 5 **hours on 8× NVIDIA V100 GPU** for the complete search process, which can be further reduced by fewer iterations. Note that for our almost distillation experiments, we directly transfer the found distiller without additional searching.

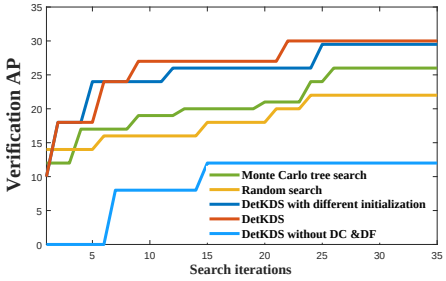

Figure 4. Search curves of different methods on RetinaNet-R50.

Scaling up to Different Students We implement our proposed Auto-DetKD on various backbones, including ResNet18, ResNet34, and ResNet50. As shown in Table 14, all the backbones obtain significant improvements in AP when used with the ResNeXt101-based RetinaNet.

5. Conclusion

In this paper, we present DetKDS framework for detection distillation search. DetKDS employs divide-and-conquer strategy and evolutionary search to navigate large search space of complex object detectors efficiently. By analyzing the search results, we derive novel insights and guidelines relevant specifically to detection distillation, such as favouring global/instance distillation, channel attention, and simple loss functions. Such guidance is valuable for future detection distillation designs. Extensive experiments evaluated various modern object detectors—two-stage, one-stage, anchor-based, and anchor-free detectors with different backbones. This breadth highlights DetKDS's applicability to diverse detection paradigms in object detection and instance segmentation. We hope that our work can inspire future research on distillation designs for dense prediction tasks.

Limitations: AutoML methods (Wang et al., 2021) all involve extra search costs but are far more efficient and automated than manual tuning. We inevitably share these same properties and make great efforts to propose acceleration strategies. In addition, our found distillers and guidance for detection distillation designs are valuable and novel.

Acknowledgements

The research was supported by Theme-based Research Scheme (T45-205/21-N) from Hong Kong RGC, and Generative AI Research and Development Centre from InnoHK.

Impact Statement

The primary focus of our DetKDS is technological advancements and improving machine learning algorithms. As such, our work does not directly involve ethical considerations or has immediate societal consequences.

References

Bolya, D., Foley, S., Hays, J., and Hoffman, J. Tide: A general toolbox for identifying object detection errors. In *ECCV*, 2020.

Cao, S., Wang, X., and Kitani, K. M. Learnable embedding space for efficient neural architecture compression. In *ICLR*, 2019.

Cao, S., Joshi, D., Gui, L., and Wang, Y.-X. HASSOD: Hierarchical adaptive self-supervised object detection. In *NeurIPS*, 2023a.

Cao, S., Joshi, D., Gui, L.-Y., and Wang, Y.-X. Contrastive mean teacher for domain adaptive object detectors. In *CVPR*, 2023b.

Cao, S., Li, M., Hays, J., Ramanan, D., Wang, Y.-X., and Gui, L. Learning lightweight object detectors via multi-teacher progressive distillation. In *ICML*, 2023c.

Cao, W., Zhang, Y., Gao, J., Cheng, A., Cheng, K., and Cheng, J. Pkd: General distillation framework for object detectors via pearson correlation coefficient. In *NeurIPS*, 2022.

Chen, W., Gong, X., Liu, X., Zhang, Q., Li, Y., and Wang, Z. Fasterseg: Searching for faster real-time semantic segmentation. In *ICLR*, 2020.

Chen, Y., Yang, T., Zhang, X., Meng, G., Xiao, X., and Sun, J. Detnas: Backbone search for object detection, 2019.

Dai, X., Jiang, Z., Wu, Z., Bao, Y., Wang, Z., Liu, S., and Zhou, E. General instance distillation for object detection. In *CVPR*, pp. 7842–7851, June 2021.

De Rijk, P., Schneider, L., Cordts, M., and Gavrila, D. Structural knowledge distillation for object detection. In *NeurIPS*, 2022.

Dong, P., Niu, X., Li, L., Xie, L., Zou, W., Ye, T., Wei, Z., and Pan, H. Prior-guided one-shot neural architecture search. *arXiv preprint arXiv:2206.13329*, 2022.

Dong, P., Li, L., and Wei, Z. Diswot: Student architecture search for distillation without training. In *CVPR*, 2023a.

Dong, P., Li, L., Wei, Z., Niu, X., Tian, Z., and Pan, H. Emq: Evolving training-free proxies for automated mixed precision quantization. In *ICCV*, 2023b.

Dong, P., Li, L., Pan, X., Wei, Z., Liu, X., Wang, Q., and Chu, X. Parzc: Parametric zero-cost proxies for efficient nas. *arXiv preprint arXiv:2402.02105*, 2024.

Girshick, R. Fast r-cnn. In *ICCV*, December 2015.

Girshick, R., Donahue, J., Darrell, T., and Malik, J. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, pp. 580–587, June 2014.

Guo, J., Han, K., Wang, Y., Wu, H., Chen, X., Xu, C., and Xu, C. Distilling object detectors via decoupled features. In *CVPR*, pp. 2154–2164, June 2021.

He, K., Gkioxari, G., Dollár, P., and Girshick, R. Mask R-CNN. In *ICCV*, pp. 2980–2988, 2017.

Heo, B., Kim, J., Yun, S., Park, H., Kwak, N., and Choi, J. Y. A comprehensive overhaul of feature distillation. In *ICCV*, pp. 1921–1930, October 2019.

Hinton, G., Vinyals, O., and Dean, J. Distilling the knowledge in a neural network (2015). *arXiv preprint arXiv:1503.02531*, 2(7), 2015.

Hu, Y., Wang, X., Li, L., and Gu, Q. Improving one-shot nas with shrinking-and-expanding supernet. *Pattern Recognition*, 2021.

Huang, T., You, S., Wang, F., Qian, C., and Xu, C. Knowledge distillation from a stronger teacher. *arXiv preprint arXiv:2205.10536*, 2022.

Huang, T., Zhang, Y., You, S., Wang, F., Qian, C., Cao, J., and Xu, C. Masked distillation with receptive tokens. In *ICLR*, 2023a.

Huang, T., Zhang, Y., Zheng, M., You, S., Wang, F., Qian, C., and Xu, C. Knowledge diffusion for distillation. *arXiv preprint arXiv:2305.15712*, 2023b.

Li, H., Tao, C., Zhu, X., Wang, X., Huang, G., and Dai, J. Auto seg-loss: Searching metric surrogates for semantic segmentation. In *ICLR*, 2021.

Li, L. Self-regulated feature learning via teacher-free feature distillation. In *ECCV (ECCV)*, 2022.

Li, L. and Jin, Z. Shadow knowledge distillation: Bridging offline and online knowledge transfer. In *NeuIPS*, 2022.

Li, L., Shiuan-Ni, L., Yang, Y., and Jin, Z. Teacher-free distillation via regularizing intermediate representation. In *IJCNN*, 2022.

Li, L., Dong, P., Li, A., Wei, Z., and Ya, Y. Kd-zero: Evolving knowledge distiller for any teacher-student pairs. In *NeurIPS*, 2023a.

Li, L., Dong, P., Wei, Z., and Yang, Y. Automated knowledge distillation via monte carlo tree search. In *ICCV (ICCV)*, 2023b.

Li, X., Wu, J., Fang, H., Liao, Y., Wang, F., and Qian, C. Local correlation consistency for knowledge distillation. In *ECCV*. Springer, 2020.

Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., and Zitnick, C. L. Microsoft coco: Common objects in context. In *ECCV*, 2014.

Lin, T.-Y., Goyal, P., Girshick, R., He, K., and Dollar, P. Focal loss for dense object detection. In *ICCV (ICCV)*, pp. 2980–2988, Oct 2017.

Lu, L., Chen, Z., Lu, X., Rao, Y., Li, L., and Pang, S. Uniads: Universal architecture-distiller search for distillation gap. In *AAAI*, 2024.

Pengguang, C., Shu, L., Hengshuang, Z., and Jia, J. Distilling knowledge via knowledge review. In *CVPR*, 2021.

Ren, S., He, K., Girshick, R., and Sun, J. Faster r-cnn: Towards real-time object detection with region proposal networks. In *NeurIPS*, 2015.

Romero, A., Ballas, N., Kahou, S. E., Chassang, A., Gatta, C., and Bengio, Y. Fitnets: Hints for thin deep nets. *arXiv preprint arXiv:1412.6550*, abs/1412.6550, 2014.

Samet, N., Hicsonmez, S., and Akbas, E. Houghnet: Integrating near and long-range evidence for bottom-up object detection. In *ECCV*, 2020.

Shu, C., Liu, Y., Gao, J., Yan, Z., and Shen, C. Channel-wise knowledge distillation for dense prediction. In *ICCV*, pp. 5311–5320, 2021.

Sun, Z., Cao, S., Yang, Y., and Kitani, K. M. Rethinking transformer-based set prediction for object detection. In *ICCV*, 2021.

Tian, Z., Shen, C., Chen, H., and He, T. Fcos: Fully convolutional one-stage object detection. In *ICCV*, October 2019.

Wang, T., Yuan, L., Zhang, X., and Feng, J. Distilling object detectors with fine-grained feature imitation. In *CVPR*, pp. 4933–4942, June 2019.

Wang, X., Cao, S., Li, M., and Kitani, K. M. Neighborhood-aware neural architecture search. *arXiv preprint arXiv:2105.06369*, 2021.

Wei, Z., Pan, H., Li, L., Dong, P., Tian, Z., Niu, X., and Li, D. Tvt: Training-free vision transformer search on tiny datasets. *arXiv preprint arXiv:2311.14337*, 2023.

Wei, Z., Dong, P., Hui, Z., Li, A., Li, L., Lu, M., Pan, H., and Li, D. Auto-prox: Training-free vision transformer architecture search via automatic proxy discovery. In *AAAI*, 2024.

Xiaolong, L., Lujun, L., Chao, L., and Yao, A. Norm: Knowledge distillation via n-to-one representation matching. In *ICLR*, 2023.

Yang, Z., Li, Z., Jiang, X., Gong, Y., Yuan, Z., Zhao, D., and Yuan, C. Focal and global knowledge distillation for detectors. In *CVPR*, pp. 4643–4652, June 2022a.

Yang, Z., Li, Z., Shao, M., Shi, D., Yuan, Z., and Yuan, C. Masked generative distillation. In *ECCV*, 2022b.

Zhang, L. and Ma, K. Improve object detection with feature-based knowledge distillation: Towards accurate and efficient detectors. In *ICLR*, 2021.

Zheng, Z., Ye, R., Wang, P., Ren, D., Zuo, W., Hou, Q., and Cheng, M.-M. Localization distillation for dense object detection. In *CVPR*, pp. 9407–9416, June 2022.

Zhixing, D., Zhang, R., Chang, M., Liu, S., Chen, T., Chen, Y., et al. Distilling object detectors with feature richness. *NeurIPS*, 34:5213–5224, 2021.

Zhu, C., Li, L., Wu, Y., and Sun, Z. Saswot: Real-time semantic segmentation architecture search without training. In *AAAI*, 2024.

A. More Ablations

A.1. More Ablation Study on Proxy settings

To accelerate the search process, we adopt a subset of the COCO train2017 dataset, known as "COCO minitrain (Samet et al., 2020)." This subset consists of 25,000 images, which is approximately 20% of the original train2017 dataset, and contains around 184,000 objects across 80 different object categories. We utilize this subset to reduce the search cost while maintaining our method's effectiveness. To validate the suitability of COCO minitrain as a representative subset, HoughNet (Samet et al., 2020) assesses the correlation between the model's performance when trained on minitrain compared to when trained on the full train2017 dataset. Figure 5 displays the object detection results obtained on train2017 and minitrain (Samet et al., 2020). Notably, the figure showcases a robust positive correlation between the results obtained from train2017 and minitrain. The Pearson correlation coefficients are 0.74 and 0.92 for the COCO evaluation metrics AP and AP50 respectively. Such strong positive correlations give us high confidence that the relative ordering of distillation configurations by minitrain score will translate accurately to their ranking according to the train2017 evaluation. This allows us to efficiently search for effective distillations without extensive evaluation of the entire dataset, significantly reducing the search cost and accelerating the overall process.

B. Result Visualization

Detection Error Analysis. The analysis of the detection error, based on the error analysis results in Figure 6 and precision-recall curves in Figure 7, demonstrates the effectiveness of our distillation method in improving the locating and classifying abilities of the student baseline detector. The findings indicate that our distillation approach leads to a reduction in various types of errors. Notably, it effectively reduces background errors, minimizes false detections, and improves the specificity of the detector. Additionally, our method addresses missed ground truth errors, enhancing the detector's sensitivity in detecting and classifying objects accurately. In summary, our distillation method significantly improves the detector's localization and classification abilities, making it more reliable and robust in object detection tasks.

Qualitative Analysis. Figure 8, Figure 9, Figure 10, and Figure 11 visualize the detection outputs of different detectors respectively. The qualitative analysis of the detection visualization results highlights several vital features exhibited by our DetKDS distillation detector compared to the student baseline detector. Firstly, our method demonstrates improved accuracy in detecting small targets, thereby enhancing the overall detection ability for such objects. This improvement is evident in the figures, where the distilled model consistently and correctly identifies cars, handbags, and people inside the car, showcasing the effectiveness of our distillation approach in accurately detecting small objects. Additionally, our DetKDS method effectively reduces false positives and instances where the detector mistakenly identifies non-existent objects. This reduction in false positives is crucial as it allows for more reliable and accurate object detection, minimizing the chances of erroneous interpretations and decisions based on incorrect detections. Moreover, our approach also addresses the issue of false negatives, where the detector fails to detect objects that are present in the image. By reducing false negatives, our distillation detector ensures that essential objects are not missed, enhancing the comprehensiveness and reliability of the detection results. Furthermore, our DetKDS method mitigates the risk of performance degradation, ensuring that the overall performance of the detector remains robust and reliable. By reducing errors and improving accuracy, our distillation approach prevents performance degradation, making the detector more effective and dependable for various applications. These findings highlight the strengths of our method and its potential for enhancing object detection performance.

C. More Discussions

C.1. Discussion on Limitations and Future Work

As a search-based distillation method, DetKDS naturally inherits similar limitations with other knowledge distillation and search methodologies (Hu et al., 2021; Dong et al., 2022; 2023b; Zhu et al., 2024; Dong et al., 2024; Wei et al., 2023; Lu et al., 2024; Wei et al., 2024). One such limitation is the potential introduction of teacher model bias and the associated cost of additional teacher model preparation. Similar to most distillation methods (Li et al., 2022; Li, 2022), our approach relies on a teacher model to transfer knowledge to the student model. However, the teacher model's biases and limitations may influence the student model's learning process, potentially impacting its performance. Preparing the teacher model requires additional computational resources and time, which can be a significant cost. Nevertheless, it is essential to note that our work can be extended to self-KD scenarions in distiller search, thereby avoiding these limitations. In self-KD, the teacher model is generated internally within the distiller search process, eliminating the need for an external teacher

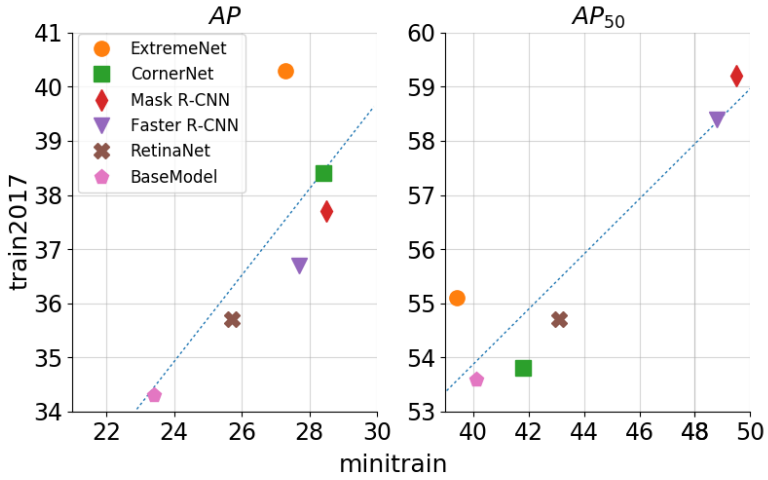

Figure 5. Correlation visualization of different detector performances in full COCO dataset and COCO minitrain (Samet et al., 2020).

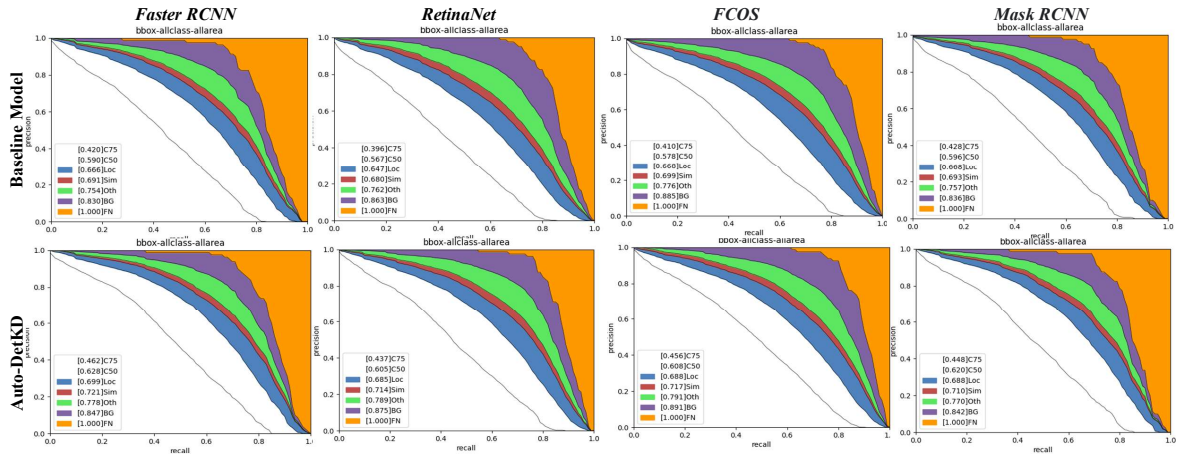

Figure 6. Error analyses (Bolya et al., 2020) of baseline students (***First Row***) and students distilled by our approach (***Second Row***) on COCO benchmarks. Cls: classification errors; Loc: localization errors; Both: both cls and loc errors; Dupe: duplicate detection errors; Bkg: background errors; Miss: missed GT errors.

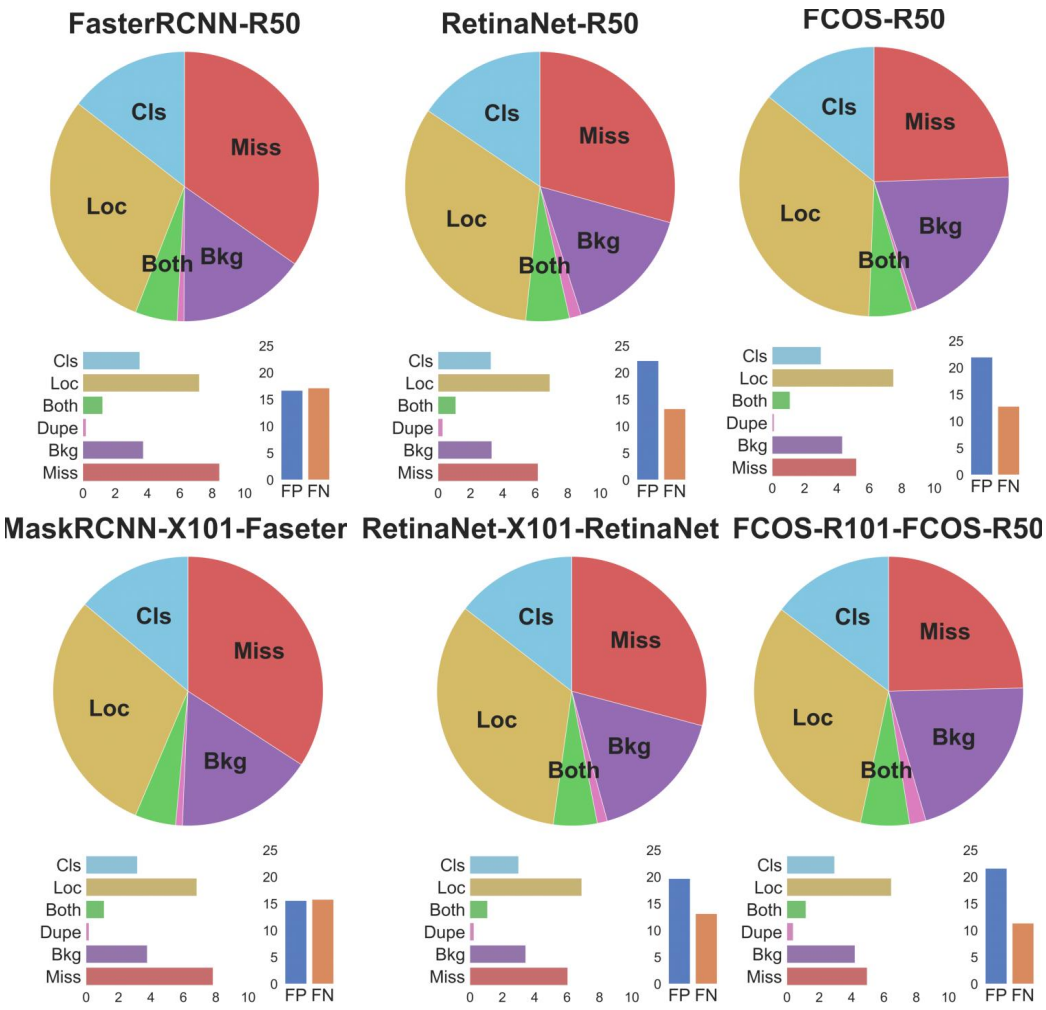

Figure 7. Precision-Recall curves of baseline students (***First Row***) and students distilled by our approach (***Second Row***) on COCO benchmarks. FN: false negative prediction; BG: Background false positive prediction; Oth: classification errors; Sim: wrong class but correct supercategory; Loc: localization errors

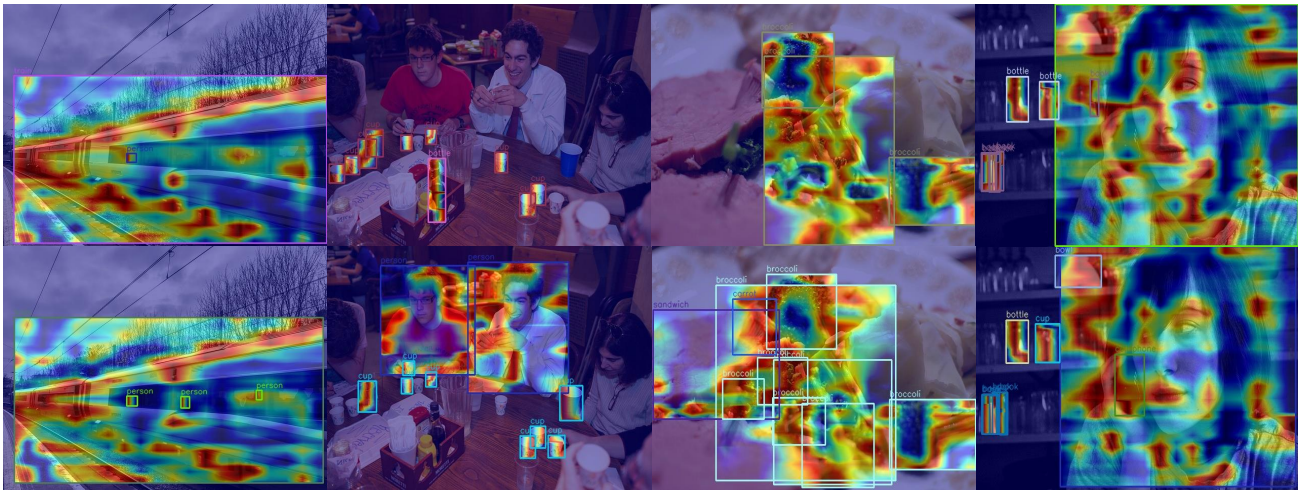

Figure 8. Qualitative analysis of baseline Faster RCNN (***First Row***) and Faster RCNN distilled by our approach (***Second Row***) on COCO benchmarks.

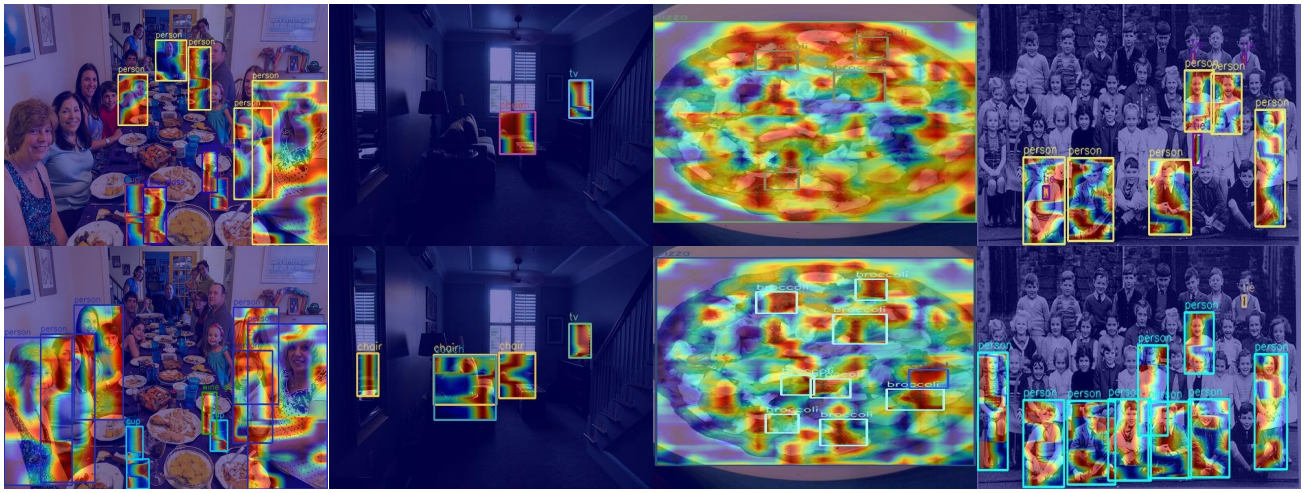

Figure 9. Qualitative analysis of baseline RetinaNet (***First Row***) and RetinaNet distilled by our approach (***Second Row***) on COCO benchmarks.

model. This mitigates teacher model bias and reduces the cost associated with its preparation. Furthermore, our approach introduces some additional overhead, as is common with many search-based methods (Li et al., 2021). The search process can be computationally intensive and time-consuming, particularly when exploring an ample search space. However, we have implemented several search acceleration and transfer strategies to alleviate this issue. These strategies aim to optimize the search process, minimize redundancy, and expedite the identification of practical distillation configurations. While our efforts address these limitations to some extent, it is crucial to recognize that certain limitations are inherent to knowledge distillation and search methodologies. In future work, we will continue refining these approaches, developing more efficient search techniques, and exploring alternative strategies to mitigate teacher model bias and reduce computational overhead. The applicability and effectiveness of search-based distillation methods like DetKDS can be further enhanced by addressing these limitations.

D. Detailed Analysis of Distiller Search Space

In this part, we present a detailed formulation and discussion of distillation operations in our search space.

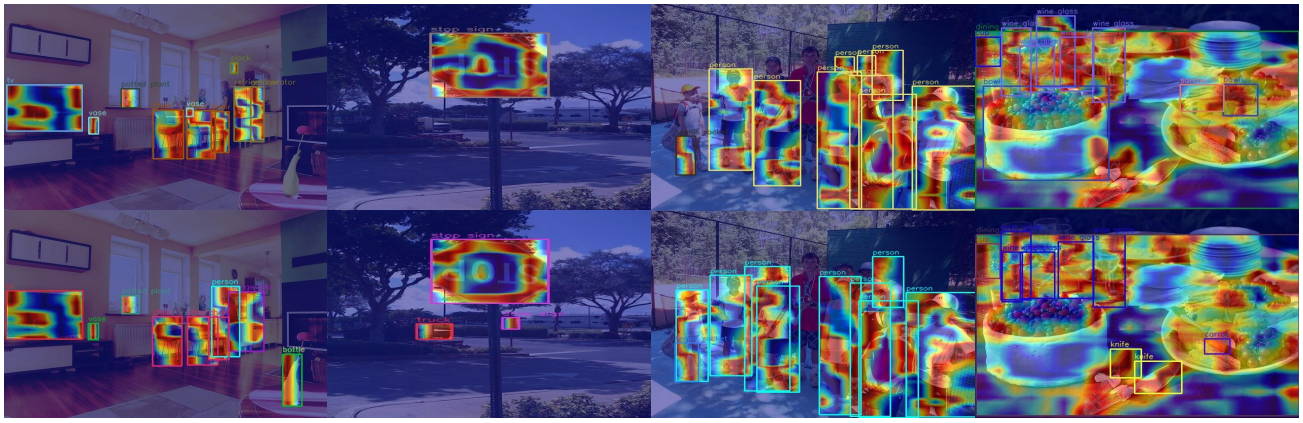

Figure 10. Qualitative analysis of baseline FCOS (***First Row***) and FCOS distilled by our approach (***Second Row***) on COCO benchmarks.

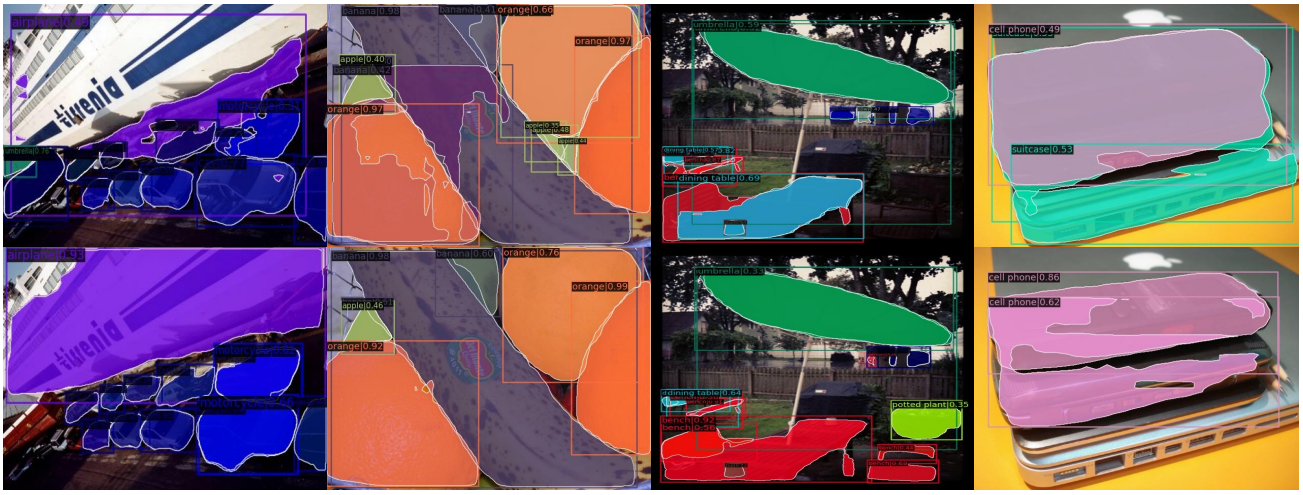

Figure 11. Qualitative analysis of baseline Mask RCNN (***First Row***) and Mask RCNN distilled by our approach (***Second Row***) on COCO benchmarks.

D.0.1. TRANSFORM OPTIONS

Channel attention transform (catt). Channel-wise features G^S and G^T are transformed by the attention transform operations as follows:

$$G^C(F) = \frac{1}{HW} \cdot \sum_{i=1}^{H} \sum_{j=1}^{W} |F_{i,j}|, A^C(F) = C \cdot \sigma(G^C(F)/\tau), \tag{9}$$

where H, W, and C denote the feature's height, width, and channel. G^S and G^C are the spatial and channel attention maps. A^S and A^C are the spatial and channel attention masks, where τ is the temperature hyperparameter to adjust the distribution.

Mask transform (mask). We use the corresponding l-th mask to cover the student's l-th feature, which can be formulated as follows:

$$M_{i,j}^l = \begin{cases} 0, & \text{if } R_{i,j}^l < \lambda \\ 1, & \text{Otherwise} \end{cases} \tag{10}$$

where $R_{i,j}^l$ is a random number in $(0, 1)$ and i, j are the horizontal and vertical coordinates of the feature map, respectively.

Scale transform (scale). Our multi-scale transformation operation extracts different levels of knowledge from the feature using spatial pyramid pooling.

Local transform (Local). We select the local transformation operation to divide the original feature into n^2 patches (*e.g.*, $n = 1, 2, 4$), then distill each patch into separate instances.

D.0.2. DISTANCE FUNCTIONS OPTIONS.

In distillation, different distance functions are used to measure the difference between teacher and student output. Let P_i denote the predicted probability of class i by the teacher network and Q_i denote the predicted probability of class i by the student network.

$\mathcal{L}_2$ distance. The $\mathcal{L}_2$ distance measures the square root of the sum of the squared differences between the probabilities of each class in the two distributions. The $\mathcal{L}_2$ distance between P and Q is defined as:

$$\ell_2 = \sqrt{\sum_{i=1}^{n}(P_i - Q_i)^2}$$

Cosine distance. The cosine distance measures the cosine of the angle between the two probability vectors. This distance measure is useful when the magnitudes of the probability vectors are not important, only their directions. The cosine distance between P and Q is defined as:

$$\ell_{Cosine} = 1 - \frac{\sum_{i=1}^{n} P_i Q_i}{\sqrt{\sum_{i=1}^{n} P_i^2}\sqrt{\sum_{i=1}^{n} Q_i^2}}$$

Pearson distance. The Pearson distance measures the correlation between the two probability vectors. The Pearson distance between P and Q is defined as:

$$\ell_{Pearson} = 1 - \frac{\sum_{i=1}^{n}(P_i - \bar{P})(Q_i - \bar{Q})}{\sqrt{\sum_{i=1}^{n}(P_i - \bar{P})^2}\sqrt{\sum_{i=1}^{n}(Q_i - \bar{Q})^2}}$$

where $\bar{P}$ and $\bar{Q}$ are the means of the two distributions.Similarly, Pearson distance is also correlated with the normalized $\mathcal{L}_2$ distance.

KL distance. The KL distance measures the information lost when approximating the probability distribution P with the probability distribution Q, as follows:

$$\ell_{KL} = \sum_{i=1}^{n} P_i \log \frac{P_i}{Q_i} = \sum_{i=1}^{n} P_i \log P_i - \sum_{i=1}^{n} P_i \log Q_i$$

Correlation distance. Let $P \in^{n \times d}$ and $Q \in^{n \times d}$ denote a batch of representations from the student and teacher, respectively. These matrices are computed before the final fully-connected layer to preserve the structural information of the data, thus enabling a strong distillation signal for the student. We first normalize these representations to zero mean and unit variance across the batch dimension and then propose to construct a cross-correlation matrix, $\mathbf{C}_{st} = P^T Q/n \in^{d \times d}$. A perfect correlation between the two sets of representations is achieved if all of the diagonal entries $v_i = (\mathbf{C}_{st})_{ii}$ are equal to one. To formulate this as a minimization problem, we propose the following loss:

$$\ell_{Cor} = \log_2 \sum_{i=1}^{d} |v_i - 1|^{2\alpha} \tag{11}$$

SSIM distance. The Structural Similarity Index (SSIM) can also be applied to compare probability distributions (De Rijk et al., 2022), such as the distributions of pixel intensities in two images. The formula for SSIM in the context of probability distributions P and Q is as follows:

$$\ell_{SSIM} = \frac{(2\mu_P \mu_Q + C_1)(2\sigma_{PQ} + C_2)}{(\mu_P^2 + \mu_Q^2 + C_1)(\sigma_P^2 + \sigma_Q^2 + C_2)}$$

where the variables μ_P and μ_Q represent the means (averages) of the probability distributions P and Q, respectively. The variables σ_P and σ_Q represent the distributions P and Q standard deviations, respectively. σ_{PQ} represents the covariance between the distributions P and Q. Similar to the image-based SSIM formula, the constants C_1 and C_2 are small positive constants added to stabilize the division and prevent division by zero. These constants are typically chosen to be small values, such as $C_1 = (k_1 \cdot L)^2$ and $C_2 = (k_2 \cdot L)^2$, where L represents the dynamic range of the probability values, and k_1 and k_2 are constants to control the impact of the means and variances. The SSIM formula for probability distributions compares the distributions' means, covariances, and variances to compute a similarity index. It measures how similar P and Q distributions are in their central tendency, spread, and relationship. Higher SSIM values indicate a higher similarity between the distributions.

IoU-Series. IoU (Intersection over Union) is a widely used metric in computer vision for evaluating the accuracy of object detection and segmentation algorithms. It measures the overlap between a predicted region and the ground truth region. By calculating the ratio of

the intersection area to the union area, IoU provides a value between 0 and 1, where higher values indicate better overlap. IoU helps researchers and practitioners objectively assess and compare different algorithms, serving as a standard benchmark in computer vision tasks. Here are the equations for IoU (Intersection over Union), C-IoU (Complete IoU), E-IoU (Exclusive IoU), G-IoU (Generalized IoU), and D-IoU (Distance-IoU) as:

1. IoU (Intersection over Union):

$$IoU = \frac{\text{Area of Intersection}}{\text{Area of Union}} = \frac{\text{Intersection} \cap \text{Union}}{\text{Intersection} \cup \text{Union}}$$

2. C-IoU (Complete IoU):

$$C - IoU = IoU - \frac{\text{Area of Enclosing Box} - \text{Area of Union}}{\text{Area of Enclosing Box}}$$

3. E-IoU (Exclusive IoU):

$$E - IoU = IoU - \frac{\text{Area of Intersection}}{\text{Area of Enclosing Box}}$$

4. G-IoU (Generalized IoU):

$$G - IoU = IoU - \alpha \cdot \frac{\text{Area of Enclosing Box} - \text{Area of Union}}{\text{Area of Enclosing Box}}$$

5. D-IoU (Distance-IoU):

$$D - IoU = IoU - \beta \cdot \frac{\text{Distance}}{\text{Diagonal Length of Enclosing Box}}$$

In these equations, "Intersection" refers to the overlapping region between two bounding boxes, "Union" represents the combined area of the two bounding boxes, "Area of Intersection" is the size of the intersecting region, "Area of Union" is the size of the union region, "Area of Enclosing Box" is the area covered by the smallest bounding box that encloses both boxes, "Distance" is the distance between the centers of the bounding boxes, and α and β are weighting parameters.