

## Appendix

### A. Effect of Different Architectures

#### A.1. Different Student

To show the generality of RDCD, we adopt one lightweight model each from the most commonly used architectures in computer vision, CNN and ViT, conducting experiments with MobileNet-V3 and FastViT-T12. Our results are shown in Table 7, Table 8, Table 9. Specifically, using FastViT under the same conditions resulted in a  $\mu\text{AP}$  of 67.4, demonstrating the research potential for ViT student.

Table 7. Performance of DISC2021 with different student architectures. \* means our implementation.

Method	Network	$dim$	$\mu\text{AP}$	$\mu\text{AP}_{SN}$
<i>using lightweight *</i>				
SSCD	Mob-V3	128	45.5	56.2
SSCD	FastViT-T12	128	42.5	59.7
SSCD	Mob-V3	256	47.3	60.1
SSCD	FastViT-T12	256	43.1	59.5
<i>ours</i>				
RDCD	Mob-V3	128	50.6	60.8
RDCD	FastViT-T12	128	48.5	61.1
RDCD	Mob-V3	256	53.9	65.6
RDCD	FastViT-T12	256	56.4	<b>67.4</b>

Table 8. Performance of CD10K(Copydays+ 10k distractors) with different student architectures. \* means our implementation.

Method	Network	$dim$	mAP	$\mu\text{AP}$
Multigrain	RN-50	1500	82.3	77.3
DINO	ViT-B/8	1536	85.3	91.7
DINO	ViT-B/16	1536	80.7	88.7
SSCD	RN-50	512	85.0	97.9
SSCD	ResNext-101	1024	91.9	96.5
<i>using lightweight *</i>				
SSCD	Mob-V3	128	68.9	97.1
SSCD	FastViT-T12	128	60.9	87.0
SSCD	Mob-V3	256	75.4	97.4
SSCD	FastViT-T12	256	74.4	92.2
<i>ours</i>				
RDCD	Mob-V3	128	81.3	96.5
RDCD	FastViT-T12	128	77.4	95.3
RDCD	Mob-V3	256	83.5	<b>97.6</b>
RDCD	FastViT-T12	256	79.3	97.0

#### A.2. Different Teacher

We demonstrate the effectiveness of our proposed approach in scenarios where the teacher networks belong to different architectural families (ViT or CNN), with student models also selected from these two distinct architectures.

Table 9. Performance of NDEC with different student architectures. \* means our implementation.

Method	Network	$dim$	$\mu\text{AP}$	$\mu\text{AP}_{SN}$
DINO	ViT-B/8	1536	16.2	22.8
DINO	ViT-B/16	1536	18.1	26.2
SSCD	RN-50	512	42.4	46.6
<i>using lightweight *</i>				
SSCD	Mob-V3	128	34.6	38.9
SSCD	FastViT-T12	128	33.5	38.6
SSCD	Mob-V3	256	36.8	41.5
SSCD	FastViT-T12	256	33.3	38.3
<i>ours</i>				
RDCD	Mob-V3	128	35.5	39.3
RDCD	FastViT-T12	128	35.1	38.7
RDCD	Mob-V3	256	37.9	42.6
RDCD	FastViT-T12	256	39.1	<b>43.2</b>

For the ViT, we use DINO as the teacher, and for the CNNs, we use SSCD-RN50 and SSCD-ResNeXt-101. All models are trained on SSCD-RN50 hyperparameter settings. With DINO as the teacher model, we follow the ICD settings presented in [5] for the DINO baseline. Consequently, we distill a descriptor size of 1536 from the teacher (concatenation of 768 class tokens and 768 GeM pooled patch tokens). In the case of ResNeXt-101 teacher model, we distill the final target descriptor with a size of 1024. Table 10 shows that our method is highly effective for the different architectures, highlighting its adaptability and robustness.

Table 10. DISC2021 evaluation with different architectures. All experiments are performed with a descriptor size of 128.

T \ S	EN-B0		FastViT-T12		Mob-V3	
	$\mu\text{AP}$	$\mu\text{AP}_{SN}$	$\mu\text{AP}$	$\mu\text{AP}_{SN}$	$\mu\text{AP}$	$\mu\text{AP}_{SN}$
DINO-ViT-B/8	32.1	52.1	28.1	52.2	39.6	54.7
SSCD-RN-50	50.0	61.1	48.5	61.1	47.3	58.8
SSCD-ResNeXt-101	44.9	59.9	42.8	57.4	41.1	56.6

### B. Intermediate Descriptor

In this study, we evaluate the performance of DISC2021 using an intermediate descriptor designed to align the descriptor size with that of the teacher model. We assess a 512-dimensional descriptor both with and without the application of HN loss, utilizing the EfficientNet-B0 network architecture. Our results (Table 12) show that the proposed RDCD method consistently outperforms SSCD across two projection sizes, 128 and 256. Furthermore, our analysis reveals that the performance gap between using and not using HN loss is negligible, and the intermediate descriptor fully exploits its 512-dimensional capacity without HN loss. This finding suggests that the student model is effectively trained under the guidance of a teacher model equipped with an entropy regularizer.

Table 11. The following table demonstrates that when dimension reduction is applied to the descriptor through PCA whitening, our approach outperforms the existing baseline in terms of  $\mu$ AP performance. We do not apply Mixup. \* means our implementation.

Method	Network	$dim$	$\mu$ AP	$\mu$ AP <sub>SN</sub>	$\mu$ AP <sub>SN256</sub>	$\mu$ AP <sub>SN128</sub>	$\mu$ AP <sub>SN64</sub>
DINO	ViT-B/8	1536	32.6	54.4	47.7	42.3	32.7
DINO	ViT-B/16	1536	32.2	53.8	47.4	42.0	32.0
SSCD	RN-50	512	43.6	72.5	66.0	56.3	38.3
SSCD	ResNext-101	1024	63.7	75.3	65.5	54.0	34.8
<i>using lightweight*</i>							
SSCD	EN-B0	64	38.2	48.5	48.5	48.5	48.5
SSCD	EN-B0	128	43.5	56.2	56.2	56.2	44.4
SSCD	EN-B0	256	46.0	59.8	59.8	53.8	42.3
SSCD	EN-B0	512	43.6	61.1	58.0	52.0	41.9
<i>Ours</i>							
RDCD	EN-B0	64	43.9	53.5	53.5	53.5	<b>53.5</b>
RDCD	EN-B0	128	50.0	61.1	61.1	61.1	53.1
RDCD	EN-B0	256	52.7	<b>65.7</b>	<b>65.7</b>	<b>61.5</b>	52.2

Table 12. Comparison of intermediate descriptor. We do not use HN loss for SSCD method as they use Koleo regularizer.

<i>128 projection</i>							
Hard Negative Loss		No		Yes			
Method	RSD	FKD	$\mu$ AP	$\mu$ AP <sub>SN</sub>	$\mu$ AP	$\mu$ AP <sub>SN</sub>	
SSCD(RN50) w/o mixup			60.4	71.1	-	-	
SSCD(EN-B0) w/o mixup			43.6	61.1	-	-	
SimCLR		✓	56.1	68.0	55.2	67.6	
	✓		56.3	69.7	56.3	69.7	
	✓	✓	56.6	69.5	56.7	69.6	
MoCo-v2		✓	51.2	67.3	54.4	67.8	
	✓		55.1	69.1	55.5	69.1	
	✓	✓	55.2	69.1	55.8	68.9	
<i>256 projection</i>							
Hard Negative Loss		No		Yes			
Method	RSD	FKD	$\mu$ AP	$\mu$ AP <sub>SN</sub>	$\mu$ AP	$\mu$ AP <sub>SN</sub>	
SSCD(RN50) w/o mixup			60.4	71.1	-	-	
SSCD(EN-B0) w/o mixup			43.6	61.1	-	-	
SimCLR		✓	56.2	68.1	55.5	67.1	
	✓		56.3	69.5	56.4	69.5	
	✓	✓	55.7	67.9	54.1	68.5	
MoCo-v2		✓	53.3	67.4	54.3	67.7	
	✓		53.7	68.2	55.4	68.9	
	✓	✓	55.8	69.2	56.1	69.0	

## C. PCA

The dimensionality of the descriptor plays a crucial role in optimizing the trade-off between matching time and accuracy during the matching phase of ICD. Previous approaches have employed PCA and PCA whitening to create compact descriptors. Following [31], the post-processing technique of PCA whitening is known for its efficacy in making the descriptor distribution more uniform. Our RDCD method, trained via deep learning, exhibits superior performance compared to the baseline, which utilizes large-sized descriptors that have undergone PCA whitening. These results are detailed in Table 11.

Table 13. Comparison of other distillation. All methods are trained with EfficientNet-B0 network. We do not use Hard Negative loss for SSCD method as they use Koleo regularizer.

<i>128 descriptors</i>							
Hard Negative Loss		No		Yes			
Method	RSD	FKD	$\mu$ AP	$\mu$ AP <sub>SN</sub>	$\mu$ AP	$\mu$ AP <sub>SN</sub>	
SSCD			43.5	56.2	-	-	
SimCLR		✓	46.2	58.7	48.4	59.9	
	✓		47.4	59.6	49.6	<u>61.4</u>	
	✓	✓	47.5	59.6	50.8	<b>61.5</b>	
MoCo-v2		✓	40.3	57.5	44.8	59.1	
	✓		47.1	60.7	50.0	<b>61.1</b>	
	✓	✓	46.2	59.3	49.6	61.0	
<i>256 descriptors</i>							
Hard Negative Loss		No		Yes			
Method	RSD	FKD	$\mu$ AP	$\mu$ AP <sub>SN</sub>	$\mu$ AP	$\mu$ AP <sub>SN</sub>	
SSCD			46.0	59.8	-	-	
SimCLR		✓	51.7	64.0	51.4	64.5	
	✓		52.2	65.6	52.7	<b>65.8</b>	
	✓	✓	52.4	65.5	49.9	64.6	
MoCo-v2		✓	41.3	56.6	46.1	60.1	
	✓		47.2	61.0	52.9	<u>65.7</u>	
	✓	✓	47.3	60.8	53.2	<b>65.9</b>	

## D. Comparison with Other Forms of Distillation

To further demonstrate the efficacy of RSD, we conduct a comparative analysis with FKD by 1) employing FKD alone, 2) employing RSD alone, and 3) integrating both FKD and RSD in Table 13. We also employ SimCLR-style contrastive learning. When FKD and RSD are combined, the best performance is observed in several cases, suggesting that combining both distillation methods can further improve performance. Nevertheless, RSD consistently outperforms FKD in all cases, with RSD alone achieving perfor-

Table 14. Ablation study on the computational cost during training. All experiments are conducted on a single A100 GPU using FP32 precision with EFF-B0 architecture and a descriptor size of 256. All metrics are calculated with a batch size of 256 except for FLOPs, which we measured with a batch size of 1.

Method	Contrastive Learning	Distillation	#Params (M)	FLOPs (G)	Time per Epoch (min)	VRAM (G)
SSCD	SimCLR	X	4.3	0.828	58.46	49.7
-	-	O	29.3	8.732	40.06	28.1
-	SimCLR	O	29.4	9.16	74.34	54.9
<b>RDCD</b>	MoCo-v2	O	34.2	9.9	51.25	29.8

Table 15. Comparison of computational cost during evaluation, highlighting our method’s high efficiency. We evaluate images with 288x288 size. FLOPs is calculated with a batch size of 1.

Method	Architecture	#Params (M)	FLOPs (G)	Throughput (image/sec)	Search Time	dim	$\mu$ AP
DINO	ViT-B/8	85.204	133.699	87.76	37.5	1536	54.4
SSCD	RN-50	24.6	8.265	903.58	15.518	512	72.5
<b>RDCD</b>	EN-B0	<b>4.8</b>	<b>0.829</b>	<b>1087.75</b>	<b>10.121</b>	256	65.7

mance on par with the combined use of both distillation methods. This indicates that RSD is sufficiently robust, and FKD does not significantly alter the outcome. It confirms that RSD can operate alone to enhance the performance of compact descriptors.

### E. Practicality of RDCD

We perform additional analysis of practicality on RDCD. To highlight the computational cost of each component, in Table 14 we conduct ablation experiments by removing contrastive learning and distillation. SSCD uses SimCLR without distillation. We found that our method requires 29.9 million more parameters than SSCD for training, primarily due to the large teacher network, which accounts for 24.0M parameters. Additionally, our method consumes a total of 9.9G of FLOPs: 8.3G for distillation and 1.6G for contrastive learning. Nevertheless, SSCD’s use of SimCLR leads to a proportional increase in VRAM usage with larger batch sizes. In our RDCD, we adopt MoCo-v2 which stores negative features in a queue, significantly reducing VRAM usage. By leveraging MoCo-v2, our method achieves 0.6 times lower VRAM usage compared to SSCD and also reduces training time. Furthermore, our final performance surpasses SSCD by 5.9.

Our RDCD incurs higher computational cost during training but shows significant efficiency during evaluation. Specifically, RDCD requires only 4.8M parameters during evaluation, which includes an 0.5M additional parameters from the EFF-B0 architecture. In copy detection, the evaluation involves two stages: model inference and database search. High-speed search is crucial when retrieving from a large-scale image database in real-world applications. We evaluate search time using the DISC2021 dataset, which consists of 1M database images and 50k query images.

As shown in Table 15, RDCD achieves superior efficiency across all metrics compared to previous methods. DINO and SSCD require 133.699G, 8.265G FLOPs, respectively, due to their large architectures, while RDCD operates with 161 times lower FLOPs than DINO and 9.9 times lower FLOPs than SSCD during model inference.

### F. Qualitative Examples

In Figure 6, we present the queries along with the top-2 results retrieved by both the RDCD and SSCD methods for cases where both methods accurately identify the ground truth. However, it is noteworthy that RDCD consistently retrieves a smaller similarity score difference between the ground truth and the hardest negative sample, compared to SSCD. For our matching evaluation, we employ a descriptor of size 256, utilizing the EN-B0 network architecture.

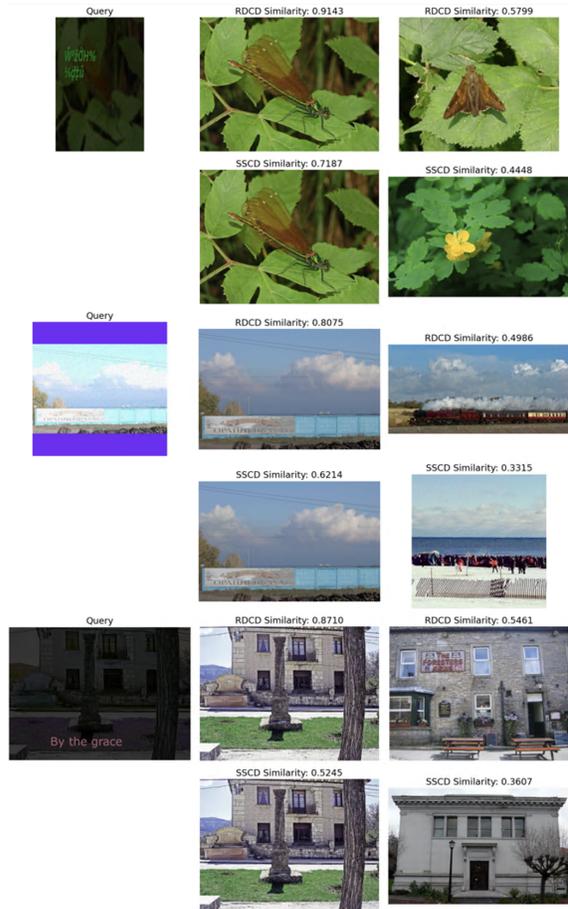


Figure 6. Example of top-2 retrieval results from the DISC2021 dataset.