Learning Compatible Multi-Prize Subnetworks for Asymmetric Retrieval

Supplementary Material

A. Additional details of weight inheritance

We briefly present our preliminary experiment in the main manuscript. Herein we provide more details and analyses. We perform pruning with the edge-popup algorithm on an 8-layer convolutional network following [10]. Specifically, we attach a learnable score to each randomly initialized weight of the network, keeping the weight frozen while updating the score to discover a good subnetwork during training. We explore two pruning strategies, One-shot Pruning (OSP) and Iterative Pruning (IP) in our preliminary experiment. OSP proposed in [10] is employed as the control group, and IP is introduced to investigate the weight inheritance nature of multi-prize subnetworks.

As presented in [10], the subnetwork discovered by OSP at a capacity of 50% achieves the best performance among all the subnetworks with various capacities. Thus, we begin with a subnetwork at the capacity of 50% to perform iterative pruning. For example, we identify a well-performing 40%-subnetwork from the 50%-subnetwork and repeat this process in a greedy pruning manner to progressively obtain subnetworks of varying capacities. As illustrated in Figure 2 in the main manuscript, the subnetworks identified by IP outperform those obtained by OSP. It empirically demonstrates that small-capacity prize subnetwork can be obtained by selectively inheriting weights from a large-capacity prize subnetwork, rather than searching for it within the entire dense network.

For the rationale behind the weight inheritance nature, we speculate that connections within a network exhibit varying degrees of importance. Integrating a set of critical connections is essential for identifying a well-performing subnetwork. The performance of a highly sparse subnetwork can be enhanced by adding an appropriate number of connections until redundancy arises. Furthermore, when attempting to directly identify a highly sparse subnetwork using the OSP method, critical connections are often excluded prematurely during the early training stages due to incomplete convergence of the learned scores. This explains why OSP tends to be less effective than the IP approach for identifying sparse subnetworks.

Algorithm 1: Training process of our method

Require: Batch input $\mathcal{B}$, the dense model ϕ_0, model parameters θ, N capacity factors $\{c_i\}_{i=1}^{N}$;

// Backward propagation

$\boldsymbol{g}_0 \leftarrow \frac{\partial \mathcal{L}_0(\phi_0, \mathcal{B})}{\partial \phi_0}$;

for $c_i \in \{c_i\}_{i=1}^{N}$ **do**

  $\phi_i \leftarrow \text{GetSubmodel}(\phi_0, c_i)$;

  $\boldsymbol{g}_i \leftarrow \frac{\partial \mathcal{L}_i(\phi_i, \mathcal{B})}{\partial \phi_i}$;

end

$G, G_{ori} \leftarrow \{\boldsymbol{g}_0, \boldsymbol{g}_1, ..., \boldsymbol{g}_N\}$;

// Conflict-aware gradient integration

for $\boldsymbol{g}_i \in G$ **do**

  $G' \leftarrow \text{Shuffle}(G)$;

  for $\boldsymbol{g}_j \in G'$ **do**

    if $\boldsymbol{g}_i \cdot \boldsymbol{g}_j < 0$ **then**

      $\boldsymbol{g}_i \leftarrow \boldsymbol{g}_i - \frac{\boldsymbol{g}_i \cdot \boldsymbol{g}_j}{\|\boldsymbol{g}_j\|^2} \boldsymbol{g}_j$;

    end

  end

end

// Calculate the cosine similarities

for $\hat{\boldsymbol{g}}_i, \boldsymbol{g}_k \in G, G_{ori}$ **do**

  $\gamma_i \leftarrow \langle \boldsymbol{g}_k, \hat{\boldsymbol{g}}_i \rangle^{\alpha}$;

end

$\tilde{\boldsymbol{g}} \leftarrow \sum \frac{\gamma_i \hat{\boldsymbol{g}}_i}{\sum \gamma_i} (N+1)$;

return Update ϕ_0 by $\Delta\theta = \tilde{\boldsymbol{g}}$

B. Additional implementation details

Training setup. We train the proposed models on two NVIDIA GeForce RTX 3090 GPUs with a batch size of 64, following the training protocols established by previous studies [4, 8, 19] on various benchmarks. On GLDv2 [15], we train Convolutional Neural Networks (CNNs), including ResNet [3], MobileNet-V2 [11], and ResNeXt [18], for 30 epochs using the Stochastic Gradient Descent (SGD) optimizer with a base learning rate of 0.1, milestones at epochs [5, 10, 20], and a weight decay of 5e-4. For ViT-Small [2], we use the AdamW optimizer, training for 30 epochs with a base learning rate of 3e-5 and a cosine decay scheduler with three epochs of linear warm-up. On the In-shop dataset [7], we optimize ResNet-18 for 200 epochs with SGD, a base learning rate of 0.1, milestones at [50, 100, 150], and a weight decay of 5e-4. On VeRi-776 [6], ResNet-18 is trained using SGD for 60 epochs with a base learning rate of 0.01, employing a Cosine Annealing Learning Rate Scheduler after the 30-th epoch.

Adaptive BatchNorm. We provide a detailed explanation of Adaptive BatchNorm [5], which is employed to address the significant discrepancy in the mean and variance of Batch Normalization (BN) layers across subnetworks of different capacities. Specifically, we set the network to training mode, freeze all learnable parameters, reset the

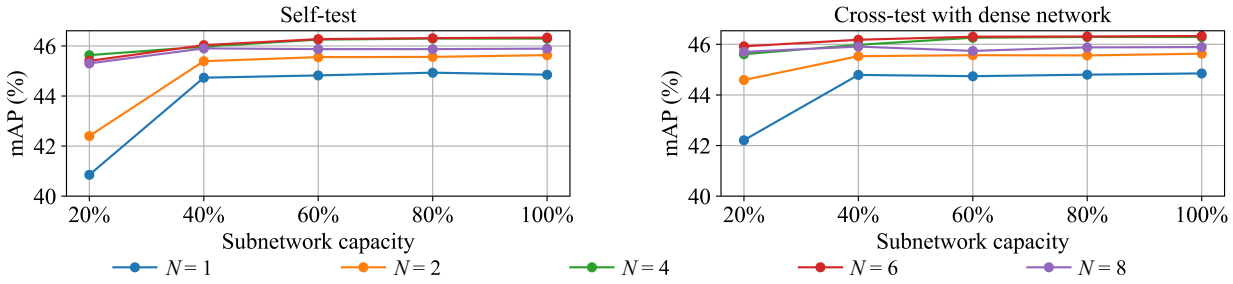

Figure A. Performance of our PrunNet when different numbers of pre-defined subnetworks are used for modeling training. We show the average mAP of RParis [9], ROxford [9], and GLDv2-test [15]. The cross-test values at 100% capacity are identical to those of the self-test.

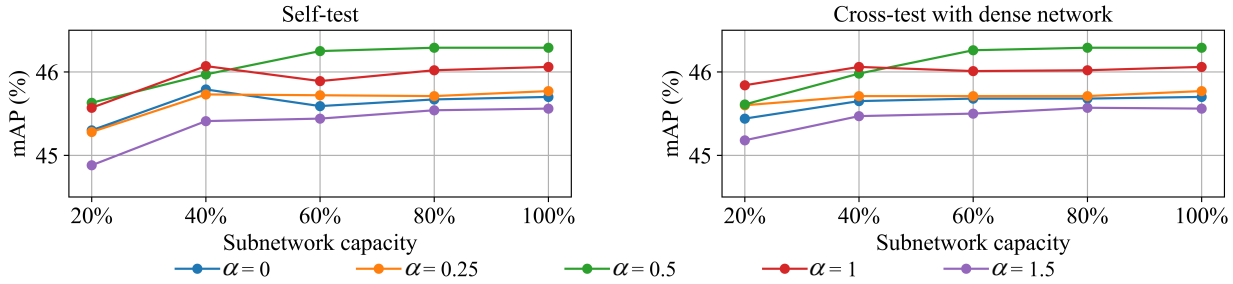

Figure B. Performance across different values of α in Eq. (5) in the main manuscript. We show the average mAP of RParis [9], ROxford [9], and GLDv2-test [15]. The cross-test values at 100% capacity are identical to those of the self-test. When α is set to 0, our method is simplified to direct gradient integration after projection.

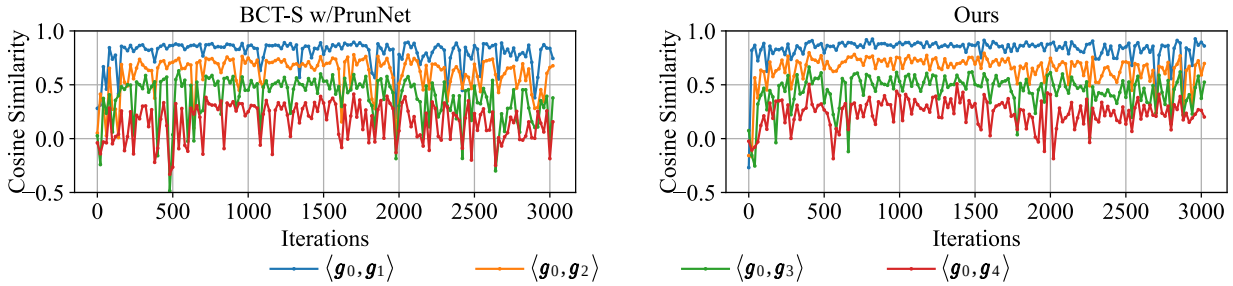

Figure C. Cosine similarities between the gradient vectors of a single convolutional kernel in the dense network and each subnetwork when training PrunNet on GLDv2 [15]. ResNet-18 is used as the backbone. Herein $\boldsymbol{g}_0$ denotes the gradient vector of a convolutional kernel of the dense network, while $\boldsymbol{g}_1$, $\boldsymbol{g}_2$, $\boldsymbol{g}_3$, and $\boldsymbol{g}_4$ represent those of the subnetworks $\phi_{80\%}$, $\phi_{60\%}$, $\phi_{40\%}$, and $\phi_{20\%}$, respectively. $\langle \cdot, \cdot \rangle$ denotes the cosine similarity. The gradient vector of each subnetwork conflicts with that of the dense network at the beginning of the training. As training progresses, negative cosine similarities in our method occur only occasionally. In contrast, the subnetworks trained with the BCT-S method encounter negative cosine similarities more frequently.

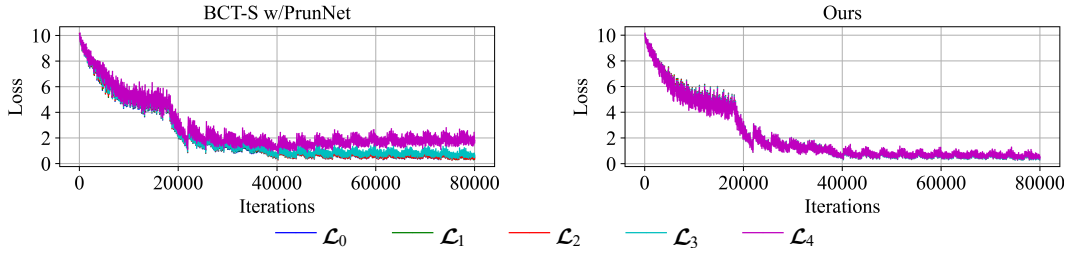

Figure D. Loss convergence curves when training PrunNet with our method and BCT-S on GLDv2 [15]. $\mathcal{L}_0$ denotes the loss of dense network ϕ_0, $\mathcal{L}_1$, $\mathcal{L}_2$, $\mathcal{L}_3$, and $\mathcal{L}_4$ denote the loss of the subnetworks $\phi_{80\%}$, $\phi_{60\%}$, $\phi_{40\%}$, and $\phi_{20\%}$, respectively. ResNet-18 is used as the backbone. The loss for both methods declines sharply at the beginning. However, as training progresses, BCT-S struggles to further reduce the losses of subnetworks. In contrast, the losses of all networks remain consistent and converge to lower values when using our method.

Table A. Detailed comparisons on pre-determined capacities over RParis [9], ROxford [9] and GLDv2-test [15]. ResNet-18 is used as the backbone. ϕ_0 denotes the dense network. The numerical subscript of a small-capacity (sub)network represents its capacity.

ϕ_q \ ϕ_g	ϕ_0	$\phi_{80\%}$	$\phi_{60\%}$	$\phi_{40\%}$	$\phi_{20\%}$	ϕ_0	$\phi_{80\%}$	$\phi_{60\%}$	$\phi_{40\%}$	$\phi_{20\%}$	ϕ_0	$\phi_{80\%}$	$\phi_{60\%}$	$\phi_{40\%}$	$\phi_{20\%}$
RParis															
	Independent learning					Joint learning					O2O-SSPL				
ϕ_0	73.35	–	–	–	–	71.58	70.94	70.72	69.88	69.11	73.35	71.94	71.50	71.08	69.40
$\phi_{80\%}$	–	71.84	–	–	–	71.53	71.50	71.08	70.14	69.49	72.16	70.71	70.19	69.85	68.25
$\phi_{60\%}$	–	–	70.71	–	–	71.22	70.66	70.75	70.58	69.69	71.91	70.58	70.26	69.74	68.26
$\phi_{40\%}$	–	–	–	70.37	–	70.65	70.28	70.10	69.28	68.56	71.96	70.59	70.18	69.89	68.14
$\phi_{20\%}$	–	–	–	–	67.77	70.79	70.24	70.05	68.80	68.80	70.19	68.94	68.32	68.14	66.72
	BCT-S w/ SwitchNet					Asymmetric-S w/ SwitchNet					SFSC				
ϕ_0	69.51	69.30	69.07	68.66	67.77	72.36	57.82	55.61	55.05	52.32	71.03	71.01	70.90	70.51	69.46
$\phi_{80\%}$	69.37	69.14	68.91	68.55	67.66	56.49	52.21	47.94	47.13	43.39	71.19	71.19	71.06	70.67	69.62
$\phi_{60\%}$	69.17	68.96	68.77	68.43	67.53	55.50	48.80	49.57	47.08	43.46	71.09	71.08	71.03	70.65	69.53
$\phi_{40\%}$	68.92	68.71	68.45	68.21	67.44	54.42	47.98	46.83	48.80	44.52	70.18	70.16	70.15	69.81	68.64
$\phi_{20\%}$	68.20	68.00	67.82	67.56	66.92	52.62	45.15	44.36	45.71	45.64	69.58	69.62	69.55	69.23	68.17
	BCT-S w/ PrunNet					Asymmetric-S w/ PrunNet					**Ours**				
ϕ_0	69.98	69.98	69.98	69.98	69.90	72.36	72.36	72.52	71.56	69.94	**74.60**	**74.59**	**74.57**	**74.53**	**74.38**
$\phi_{80\%}$	69.89	70.02	69.98	69.98	69.89	72.34	72.36	72.50	71.55	69.97	**74.62**	**74.62**	**74.60**	**74.55**	**74.40**
$\phi_{60\%}$	69.98	69.98	70.01	69.98	69.90	72.17	72.16	72.29	71.37	69.70	**74.65**	**74.64**	**74.61**	**74.58**	**74.44**
$\phi_{40\%}$	70.01	70.01	70.01	70.02	69.94	71.27	71.26	71.36	70.53	68.99	**74.53**	**74.52**	**74.50**	**74.47**	**74.31**
$\phi_{20\%}$	69.94	69.94	69.94	69.94	69.88	70.00	70.01	70.07	69.33	68.51	**74.35**	**74.35**	**74.31**	**74.28**	**74.18**
ROXford															
	Independent learning					Joint learning					O2O-SSPL				
ϕ_0	52.28	–	–	–	–	50.23	50.02	50.28	48.34	48.67	52.28	49.24	49.48	48.61	46.29
$\phi_{80\%}$	–	51.94	–	–	–	48.57	49.47	49.28	47.46	48.03	50.51	46.20	47.49	46.73	44.40
$\phi_{60\%}$	–	–	51.00	–	–	48.97	49.51	50.17	47.90	48.61	50.15	45.67	47.44	46.10	43.82
$\phi_{40\%}$	–	–	–	50.26	–	48.15	49.10	49.70	48.69	47.49	49.64	46.71	46.85	46.34	43.93
$\phi_{20\%}$	–	–	–	–	49.32	48.20	48.59	49.90	46.76	48.30	48.82	44.66	46.14	44.48	43.98
	BCT-S w/ SwitchNet					Asymmetric-S w/ SwitchNet					SFSC				
ϕ_0	52.51	**52.75**	52.02	50.89	48.97	51.90	36.31	36.37	36.97	32.67	52.59	52.40	51.71	50.86	49.35
$\phi_{80\%}$	52.49	**53.51**	**52.98**	50.66	48.79	40.36	34.69	29.60	30.26	26.99	52.31	51.96	51.37	50.90	49.30
$\phi_{60\%}$	**52.60**	**53.16**	**52.82**	50.22	48.11	39.62	31.46	32.80	30.40	27.75	51.24	51.77	51.67	51.17	49.65
$\phi_{40\%}$	51.47	51.46	51.46	51.08	51.28	37.98	30.83	31.05	33.28	28.22	51.74	51.48	51.22	50.36	48.83
$\phi_{20\%}$	50.67	50.57	50.73	49.00	46.97	37.30	28.28	29.83	31.54	29.60	50.98	50.94	50.44	49.77	48.12
	BCT-S w/ PrunNet					Asymmetric-S w/ PrunNet					**Ours**				
ϕ_0	51.54	51.53	51.54	51.14	51.31	51.80	51.88	51.60	51.29	49.28	**52.69**	52.68	**52.73**	**52.68**	**52.38**
$\phi_{80\%}$	51.54	51.54	51.57	51.14	51.30	52.25	52.33	52.12	51.43	49.21	**52.67**	52.66	52.64	**52.64**	**52.38**
$\phi_{60\%}$	51.55	51.54	51.51	51.13	51.28	51.21	51.28	51.35	51.07	48.66	52.59	52.61	52.59	**52.65**	**52.43**
$\phi_{40\%}$	51.47	51.46	51.46	51.08	51.28	50.78	50.80	50.83	49.64	48.43	**51.99**	**51.99**	**51.91**	**51.95**	**51.76**
$\phi_{20\%}$	**51.27**	**51.28**	51.26	51.13	51.29	49.52	50.10	49.78	49.40	47.34	51.19	51.22	**51.27**	**51.63**	**51.49**
GLDv2-test															
	Independent learning					Joint learning					O2O-SSPL				
ϕ_0	10.59	–	–	–	–	10.02	9.70	9.31	8.85	8.92	10.59	9.92	10.00	9.86	9.23
$\phi_{80\%}$	–	10.39	–	–	–	9.72	9.95	9.30	8.82	9.01	9.94	9.60	9.62	9.47	8.84
$\phi_{60\%}$	–	–	9.94	–	–	9.55	9.54	9.59	8.85	8.98	9.72	9.30	9.58	9.30	8.67
$\phi_{40\%}$	–	–	–	9.58	–	9.28	9.00	8.72	8.74	8.47	9.29	8.96	8.97	9.07	8.36
$\phi_{20\%}$	–	–	–	–	8.23	8.74	8.71	8.13	8.01	8.47	8.77	8.55	8.61	8.64	8.38
	BCT-S w/ SwitchNet					Asymmetric-S w/ SwitchNet					SFSC				
ϕ_0	9.29	9.20	9.03	8.85	8.58	11.00	5.21	5.01	5.10	3.64	9.79	9.75	9.43	9.25	8.54
$\phi_{80\%}$	9.22	9.19	9.04	8.84	8.55	4.32	4.26	3.20	2.87	2.11	9.70	9.69	9.28	9.08	8.50
$\phi_{60\%}$	9.08	9.03	8.99	8.79	8.53	3.86	2.96	3.87	2.93	2.00	9.48	9.38	9.04	8.91	8.38
$\phi_{40\%}$	8.84	8.79	8.75	8.74	8.43	3.48	2.59	2.74	3.42	2.10	9.08	9.07	8.80	8.78	8.30
$\phi_{20\%}$	8.12	8.15	8.11	8.10	8.22	2.63	2.06	2.07	2.42	2.55	8.45	8.35	8.14	8.05	8.00
	BCT-S w/ PrunNet					Asymmetric-S w/ PrunNet					**Ours**				
ϕ_0	9.59	9.60	9.63	9.62	9.56	11.36	11.38	11.15	10.72	9.61	**11.59**	**11.60**	**11.60**	**11.60**	**11.48**
$\phi_{80\%}$	9.61	9.61	9.63	9.62	9.56	11.32	11.51	11.18	10.72	9.63	**11.57**	**11.59**	**11.60**	**11.59**	**11.44**
$\phi_{60\%}$	9.60	9.61	9.62	9.63	9.59	11.13	11.23	11.01	10.71	9.55	**11.54**	**11.54**	**11.56**	**11.55**	**11.37**
$\phi_{40\%}$	9.64	9.65	9.65	9.67	9.59	10.34	10.38	10.57	10.40	9.39	**11.41**	**11.45**	**11.43**	**11.49**	**11.38**
$\phi_{20\%}$	9.55	9.55	9.55	9.57	9.53	9.02	9.20	9.30	9.19	8.89	**11.30**	**11.32**	**11.30**	**11.30**	**11.22**

Table B. Detailed comparisons on pre-determined capacities over RParis [9], ROxford [9] and GLDv2-test [15] using different backbones.

		$\mathcal{M}(\phi_0,\phi_0)$ Self-test	$\mathcal{M}(\phi_{80\%},\phi_{80\%})$ Self-test	$\mathcal{M}(\phi_{80\%},\phi_0)$ Cross-test	$\mathcal{M}(\phi_{60\%},\phi_{60\%})$ Self-test	$\mathcal{M}(\phi_{60\%},\phi_0)$ Cross-test	$\mathcal{M}(\phi_{40\%},\phi_{40\%})$ Self-test	$\mathcal{M}(\phi_{40\%},\phi_0)$ Cross-test	$\mathcal{M}(\phi_{20\%},\phi_{20\%})$ Self-test	$\mathcal{M}(\phi_{20\%},\phi_0)$ Cross-test
RParis										
ResNet-50	Independent learning	74.33	73.94	–	73.75	–	73.44	–	72.82	–
	SFSC	74.59	74.48	74.52	74.32	74.43	74.25	74.36	73.61	74.04
	Ours	**75.05**	**75.01**	**75.02**	**74.95**	**74.96**	**74.90**	**74.91**	**74.78**	**74.93**
ResNeXt-50	Independent learning	75.22	75.03	–	74.63	–	73.77	–	70.71	–
	SFSC	74.92	73.80	73.78	73.67	73.71	72.50	73.16	71.22	73.08
	Ours	**76.03**	**75.97**	**75.97**	**75.94**	**75.90**	**75.77**	**75.80**	**75.07**	**75.36**
MobileNet-V2	Independent learning	66.60	65.76	–	65.05	–	64.51	–	63.68	–
	SFSC	66.38	65.91	66.10	65.75	66.08	65.27	65.81	63.83	65.09
	Ours	**67.15**	**67.10**	**67.08**	**66.95**	**67.05**	**66.53**	**66.84**	**64.57**	**66.01**
ViT-Small	Independent learning	80.81	73.40	–	70.87	–	64.61	–	52.93	–
	SFSC	77.37	74.42	75.28	70.72	73.02	68.66	72.76	55.15	63.83
	Ours	**82.00**	**80.99**	**81.22**	**80.54**	**80.72**	**77.74**	**78.73**	**72.22**	**74.24**
ROxford										
ResNet-50	Independent learning	54.70	54.56	–	54.14	–	54.20	–	50.90	–
	SFSC	53.84	53.75	53.73	53.35	53.62	53.22	53.26	52.88	53.19
	Ours	**56.12**	**55.81**	**55.97**	**55.71**	**55.98**	**55.38**	**55.84**	**54.69**	**55.52**
ResNeXt-50	Independent learning	55.38	53.73	–	52.61	–	52.16	–	50.87	–
	SFSC	54.57	54.06	53.52	53.06	53.09	52.40	53.21	50.31	52.40
	Ours	**57.63**	**57.73**	**57.75**	**57.82**	**57.76**	**58.27**	**57.96**	**56.73**	**57.45**
MobileNet-V2	Independent learning	46.60	45.91	–	45.62	–	44.39	–	43.88	–
	SFSC	46.84	45.64	45.07	46.44	46.56	45.51	46.07	43.22	43.90
	Ours	**47.63**	**47.88**	**47.64**	**48.09**	**47.83**	**47.17**	**47.55**	**45.20**	**46.71**
ViT-Small	Independent learning	59.88	54.45	–	46.41	–	37.22	–	28.58	–
	SFSC	56.10	52.25	54.98	48.24	54.68	43.60	48.68	31.05	37.85
	Ours	**60.11**	**55.36**	**55.46**	**54.84**	**56.01**	**52.24**	**52.70**	**43.96**	**46.50**
GLDv2-test										
ResNet-50	Independent learning	12.15	12.03	–	11.52	–	11.38	–	10.50	–
	SFSC	10.84	10.89	11.01	10.91	11.02	10.86	10.90	9.96	10.30
	Ours	**12.46**	**12.41**	**12.38**	**12.49**	**12.43**	**12.46**	**12.45**	**12.18**	**12.39**
ResNeXt-50	Independent learning	12.92	11.95	–	11.54	–	11.41	–	10.05	–
	SFSC	11.77	11.23	11.42	10.83	11.14	10.43	10.86	9.19	9.99
	Ours	**13.03**	**13.05**	**13.01**	**13.11**	**13.02**	**12.98**	**12.99**	**12.84**	**12.92**
MobileNet-V2	Independent learning	8.38	7.94	–	7.65	–	6.65	–	6.30	–
	SFSC	7.50	7.42	7.56	7.31	7.26	6.78	7.03	6.23	6.48
	Ours	**8.80**	**8.82**	**8.82**	**8.83**	**8.78**	**8.47**	**8.70**	**7.13**	**7.77**
ViT-Small	Independent learning	15.03	12.28	–	8.89	–	5.43	–	2.96	–
	SFSC	13.40	11.01	11.90	9.71	10.68	6.89	8.19	3.33	4.35
	Ours	**15.06**	**13.96**	**14.26**	**14.01**	**14.29**	**12.18**	**12.71**	**7.45**	**9.47**

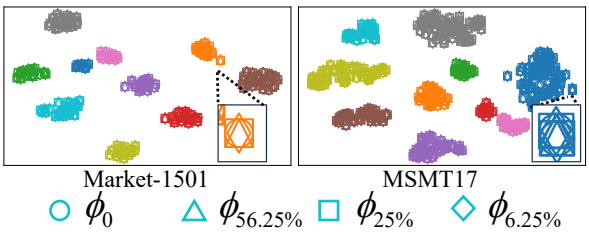

Figure E. Feature distributions of different capacities of subnetworks on Market-1501 and MSMT17 datasets visualized with t-SNE. Herein we randomly sample ten different persons on each dataset. We can observe that the feature distributions of subnetworks are aligned with that of the dense network, validating the compatibility among subnetworks.

mean and variance of BN layers to zero, and perform forward propagation using a subset of the training dataset to compute the updated statistics after training. The amounts of data used for Adaptive BatchNorm are as follows: for GLDv2, 1/30 of the training dataset is utilized, while for InShop and VeRi-776, the entire training dataset is used.

C. Pseudo algorithm

We provide the algorithm description of the optimization process in Algorithm 1.

D. More analysis and discussions

Additional analyses of hyperparameter N. We conducted additional analytical experiments to evaluate the impact of the pre-defined number of subnetworks, N, on model training, as illustrated in Figure A. For $N \leq 6$, both the dense network and the subnetworks show improved

Table C. Detailed comparisons on the new capacity (10%) over RParis [9], ROxford [9], and GLDv2-test [15]. ResNet-18 is used as the backbone. For methods without PrunNet, we use BCT [12] or SSPL [16] to train a new small-capacity model, whose capacity is 10% of the dense network ϕ_0, with compatibility with existing models.

Methods	$\mathcal{M}(\phi_{10\%}, \phi_0)$	$\mathcal{M}(\phi_{10\%}, \phi_{80\%})$	$\mathcal{M}(\phi_{10\%}, \phi_{60\%})$	$\mathcal{M}(\phi_{10\%}, \phi_{40\%})$	$\mathcal{M}(\phi_{10\%}, \phi_{20\%})$	$\mathcal{M}(\phi_{10\%}, \phi_{10\%})$
RParis						
Joint learning + BCT	70.27	69.84	69.62	68.87	68.48	68.25
O2O-SSPL +SSPL	68.83	67.62	67.09	66.90	65.33	64.30
BCT-S w/ SwitchNet + BCT	67.99	67.82	67.61	67.18	66.57	66.07
Asymmetric-S w/ SwitchNet + BCT	68.86	56.88	54.66	54.10	51.29	67.07
SFSC + BCT	68.71	68.53	68.61	68.32	67.43	66.71
BCT-S w/ PrunNet	68.79	68.79	68.81	68.80	68.67	65.93
Asymmetric-S w/ PrunNet	62.27	62.21	62.37	61.94	61.63	56.09
Ours	**73.42**	**73.41**	**73.40**	**73.41**	**73.38**	**70.12**
ROxford						
Joint learning + BCT	**50.65**	**50.55**	48.73	49.03	48.68	47.48
O2O-SSPL + SSPL	47.95	45.24	45.82	44.92	42.10	43.05
BCT-S w/ SwitchNet +BCT	49.39	49.38	49.10	47.72	45.36	46.22
Asymmetric-S w/ SwitchNet + BCT	50.38	34.81	35.05	35.08	31.37	47.71
SFSC + BCT	48.84	48.95	48.57	47.40	45.09	44.49
BCT-S w/ PrunNet	49.30	49.28	49.27	49.20	49.18	47.03
Asymmetric-S w/ PrunNet	46.27	46.74	46.48	46.84	44.88	42.48
Ours	50.53	50.42	**50.54**	**50.67**	**50.40**	**48.41**
GLDv2-test						
Joint learning + BCT	8.47	8.78	8.39	8.36	8.17	8.11
O2O-SSPL + SSPL	7.68	7.69	7.82	7.62	7.29	6.95
BCT-S w/ SwitchNet + BCT	7.75	7.72	7.68	7.47	7.41	7.73
Asymmetric-S w/SwitchNet + BCT	8.48	4.40	4.05	4.39	3.19	8.27
SFSC + BCT	7.23	7.14	6.93	7.03	6.84	7.47
BCT-S w/ PrunNet	8.21	8.19	8.20	8.23	8.28	8.01
Asymmetric-S w/ PrunNet	4.20	4.24	4.34	4.30	4.62	4.09
Ours	**10.07**	**10.05**	**10.04**	**10.08**	**9.87**	**9.12**

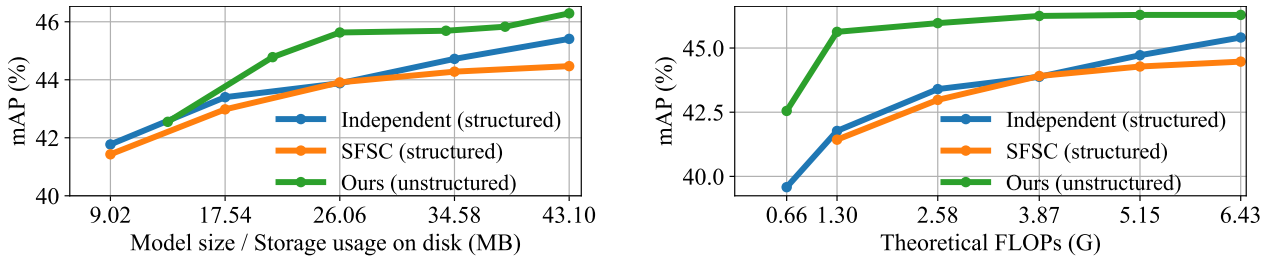

Figure F. Comparison of mAP with subnetworks of different model sizes (storage usages on disk) and theoretical FLOPs.

performance with increasing N, indicating that optimizing more subnetworks jointly benefits learning more accurate rankings of the connections. However, as N continues to increase, performance starts to degrade. This decline can be attributed to the increased difficulty in optimizing PrunNet, particularly due to the more intractable gradient conflicts arising from the larger number of subnetworks.

Analyses of the hyperparameter α. As presented in Eq. (5) in the main manuscript, we employ a hyperparameter α to control the influence of the conflicting degree on the weight. We conducted experiments to analyze the effect of α. Notably, when α is set to 0, the method is simplified to direct gradient integration after projection. Figure B illustrates the self-test and cross-test performance across different values of α. The results indicate that the best performance is achieved at $\alpha = 0.5$. Setting α to a large value causes the optimization to be dominated by gradients with minimal conflict, which hinders the effective convergence of the other subnetworks and results in degraded performance. Consequently, we set α to 0.5 for all experiments.

More visualizations. We visualize the cosine similarities between the gradient vectors of a single convolutional kernel in the dense network and each subnetwork, as shown in Figure C. We can observe that the gradient vector of each subnetwork conflicts with that of the dense network at the beginning of the training, evidenced by the negative cosine similarity. As training progresses, negative cosine similarities in our method occur only occasionally and are primarily observed in the smallest subnetwork, *i.e.* $\phi_{20\%}$. In contrast, the subnetworks trained with the BCT-S method encounter

Table D. Detailed results of different variants over RParis [9], ROxford [9] and GLDv2-test [15]. ResNet-18 is used as the backbone.

	$\mathcal{M}(\phi_0, \phi_0)$ Self-test	$\mathcal{M}(\phi_{80\%}, \phi_{80\%})$ Self-test	$\mathcal{M}(\phi_{80\%}, \phi_0)$ Cross-test	$\mathcal{M}(\phi_{60\%}, \phi_{60\%})$ Self-test	$\mathcal{M}(\phi_{60\%}, \phi_0)$ Cross-test	$\mathcal{M}(\phi_{40\%}, \phi_{40\%})$ Self-test	$\mathcal{M}(\phi_{40\%}, \phi_0)$ Cross-test	$\mathcal{M}(\phi_{20\%}, \phi_{20\%})$ Self-test	$\mathcal{M}(\phi_{20\%}, \phi_0)$ Cross-test
RParis									
Independent learning	73.35	71.84	–	70.71	–	70.37	–	67.77	–
Ours (N = 4)	**74.60**	**74.62**	**74.62**	**74.61**	**74.65**	**74.47**	**74.53**	**74.18**	**74.35**
Frozen scores	72.72	72.66	72.72	72.61	72.76	72.18	72.72	69.72	71.37
N score maps	72.01	71.45	71.57	71.38	71.69	70.88	71.38	69.57	70.99
Direct gradient integration	73.09	73.06	73.09	73.07	73.08	73.90	73.10	72.64	72.79
Direct loss combination	69.51	69.14	69.37	68.77	69.17	68.21	68.92	66.92	68.20
Pareto integration	72.10	72.09	72.09	72.11	72.09	72.04	72.10	71.36	71.71
Ours (N = 1)	72.33	72.36	72.33	72.35	72.32	72.21	72.23	67.11	70.28
Ours (N = 2)	73.56	73.45	73.49	73.39	73.41	73.31	73.42	70.32	72.31
Ours (N = 6)	73.99	74.02	74.01	73.98	73.99	73.65	73.81	72.96	73.33
Ours (N = 8)	73.58	73.56	73.58	73.52	73.56	73.47	73.54	72.76	73.14
ROxford									
Independently learning	52.28	51.94	–	51.00	–	50.26	–	49.32	–
Ours (N=4)	52.69	52.66	52.67	52.59	52.59	51.95	51.99	51.49	51.19
Frozen scores	52.03	51.86	51.95	51.74	51.80	51.78	51.87	50.08	50.80
N score maps	50.11	49.87	49.56	49.37	48.79	48.73	49.41	48.02	48.73
Direct gradient integration	52.53	52.49	52.50	52.22	52.48	52.12	52.49	52.04	52.25
Direct loss combination	51.54	51.54	51.54	51.54	51.55	51.46	51.47	51.29	51.27
Pareto integration	51.85	51.73	51.84	51.70	51.82	51.44	51.73	49.97	51.38
Ours (N = 1)	51.20	51.36	51.02	51.16	50.88	51.22	51.29	46.37	46.81
Ours (N = 2)	52.00	51.87	51.87	51.87	51.94	51.70	51.93	47.43	51.24
Ours (N = 6)	**53.82**	**53.75**	**53.76**	**53.67**	**53.74**	**53.42**	**53.62**	**52.32**	**53.37**
Ours (N = 8)	52.63	52.58	52.60	52.68	52.66	52.83	52.82	52.14	52.93
GLDv2-test									
Independently learning	10.59	10.39	–	9.94	–	9.58	–	8.23	
Ours (N=4)	**11.59**	**11.59**	**11.57**	**11.56**	**11.54**	**11.49**	**11.41**	**11.22**	**11.30**
Frozen scores	10.95	10.81	10.86	10.69	10.71	10.12	10.42	9.21	9.71
N score maps	10.66	10.57	10.59	10.18	10.39	10.06	10.37	9.11	9.43
Direct gradient integration	11.48	11.47	11.45	11.47	11.47	11.35	11.36	11.21	11.28
Direct loss combination	9.59	9.61	9.61	8.99	9.08	8.74	8.84	8.22	8.12
Pareto integration	10.57	10.57	10.58	10.58	10.58	10.62	10.63	10.23	10.39
Ours (N = 1)	11.03	11.06	11.06	10.95	11.03	10.77	10.85	9.07	9.55
Ours (N = 2)	11.33	11.37	11.31	11.39	11.36	11.16	11.24	9.46	10.22
Ours (N = 6)	11.18	11.17	11.17	11.16	11.16	11.01	11.10	10.91	11.07
Ours (N = 8)	11.45	11.47	11.47	11.40	11.41	11.39	11.36	11.01	11.02

Table E. Comparisons on pre-determined capacities over Market-1501 [20]. We employ ResNet-18 as the backbone. We use the same setting for the subnetwork capacities as SFSC [17] to include the results reported by [17] (denoted by †) in the comparison on Market-1501.

	$\mathcal{M}(\phi_0, \phi_0)$ Self-test	$\mathcal{M}(\phi_{56.25\%}, \phi_{56.25\%})$ Self-test	$\mathcal{M}(\phi_{56.25\%}, \phi_0)$ Cross-test	$\mathcal{M}(\phi_{25\%}, \phi_{25\%})$ Self-test	$\mathcal{M}(\phi_{25\%}, \phi_0)$ Cross-test	$\mathcal{M}(\phi_{6.25\%}, \phi_{6.25\%})$ Self-test	$\mathcal{M}(\phi_{6.25\%}, \phi_0)$ Cross-test
Independent learning	80.91	71.25	–	67.48	–	55.25	–
SFSC†	81.43	72.06	77.26	70.74	76.37	58.19	69.43
Ours	**81.55**	**81.25**	**81.36**	**81.32**	**81.28**	**80.08**	**80.31**

Table F. Comparisons on pre-determined capacities over MSMT17 [14]. We employ ResNet-18 as the backbone. We use the same setting for the subnetwork capacities as SFSC [17] to include the results reported by [17] (denoted by †) in the comparison on MSMT17.

	$\mathcal{M}(\phi_0, \phi_0)$ Self-test	$\mathcal{M}(\phi_{56.25\%}, \phi_{56.25\%})$ Self-test	$\mathcal{M}(\phi_{56.25\%}, \phi_0)$ Cross-test	$\mathcal{M}(\phi_{25\%}, \phi_{25\%})$ Self-test	$\mathcal{M}(\phi_{25\%}, \phi_0)$ Cross-test	$\mathcal{M}(\phi_{6.25\%}, \phi_{6.25\%})$ Self-test	$\mathcal{M}(\phi_{6.25\%}, \phi_0)$ Cross-test
Independent learning	43.06	30.06	–	22.86	–	11.69	–
SFSC†	43.89	–	37.74	–	35.32	–	28.16
Ours	**44.73**	**43.93**	**44.26**	**42.77**	**43.58**	**41.29**	**42.75**

negative cosine similarities more frequently. This indicates that our method is more effective in alleviating gradient conflicts. Besides, we observe lower cosine similarities in the sparser subnetworks, which can be attributed to the fact that they share less weight with the dense network.

We also visualize the loss convergence curves of our

Table G. Recall@1 on CUB-200 [13]. We employ ViT-S as the backbone. All models are pretrained on ImageNet-1k before being fine-tuned on CUB-200.

	$\mathcal{M}(\phi_0,\phi_0)$ Self-test	$\mathcal{M}(\phi_{80\%},\phi_{80\%})$ Self-test	$\mathcal{M}(\phi_{80\%},\phi_0)$ Cross-test	$\mathcal{M}(\phi_{60\%},\phi_{60\%})$ Self-test	$\mathcal{M}(\phi_{60\%},\phi_0)$ Cross-test	$\mathcal{M}(\phi_{40\%},\phi_{40\%})$ Self-test	$\mathcal{M}(\phi_{40\%},\phi_0)$ Cross-test	$\mathcal{M}(\phi_{20\%},\phi_{20\%})$ Self-test	$\mathcal{M}(\phi_{20\%},\phi_0)$ Cross-test
Independent learning	80.00	78.89	–	78.43	–	78.18	–	77.61	–
SFSC	80.54	80.43	80.55	80.27	80.41	80.15	80.24	78.79	78.68
Ours	**82.46**	**82.45**	**82.53**	**82.29**	**82.41**	**81.57**	**81.72**	**79.32**	**79.91**

Table H. Comparison on PrunNet implemented by structured pruning (Str.) and unstructured pruning (UnStr.) on Landmark datasets (Average mAP) and Inshop dataset (Recall@1). We employ ResNet-18 as the backbone.

		$\mathcal{M}(\phi_0,\phi_0)$ Self-test	$\mathcal{M}(\phi_{80\%},\phi_{80\%})$ Self-test	$\mathcal{M}(\phi_{80\%},\phi_0)$ Cross-test	$\mathcal{M}(\phi_{60\%},\phi_{60\%})$ Self-test	$\mathcal{M}(\phi_{60\%},\phi_0)$ Cross-test	$\mathcal{M}(\phi_{40\%},\phi_{40\%})$ Self-test	$\mathcal{M}(\phi_{40\%},\phi_0)$ Cross-test	$\mathcal{M}(\phi_{20\%},\phi_{20\%})$ Self-test	$\mathcal{M}(\phi_{20\%},\phi_0)$ Cross-test
Landmark	SFSC	44.47	44.28	44.40	43.91	43.94	42.98	43.67	41.43	43.00
	Ours (Str.)	44.81	44.72	45.04	44.46	45.14	44.07	44.33	41.58	43.10
	Ours (UnStr.)	**46.29**	**46.29**	**46.29**	**46.25**	**46.26**	**45.97**	**45.98**	**45.63**	**45.61**
In-shop	SFSC	84.57	84.48	84.40	84.25	84.31	84.15	84.20	83.57	83.74
	Ours (Str.)	86.90	86.69	86.78	86.59	86.70	86.37	86.66	86.19	86.34
	Ours (UnStr.)	**87.31**	**87.30**	**87.33**	**87.21**	**87.23**	**87.14**	**87.15**	**86.43**	**86.77**

method and BCT-S on GLDv2, as shown in Figure D. At the beginning of training, the losses for both methods decline sharply. However, as training progresses, BCT-S struggles to decrease the losses of subnetworks further. The losses of subnetworks exhibit substantial inconsistency with that of the dense network. In contrast, when training PrunNet with our method, the losses of all networks remain consistent and converge to lower values.

We show additional visualization of feature distributions across the dense network and different capacities of subnetworks in Figure E. All subnetworks exhibit feature distributions consistent with the dense network on Market-1501 [20] and MSMT17 [14] datasets, demonstrating the effectiveness of our proposed method.

Better performance than independent learning. In our proposed algorithm, the compatible losses $\mathcal{L}_1, \mathcal{L}_2, \ldots, \mathcal{L}_N$ can be interpreted as regularization terms applied to the dense network. These regularization terms are designed to encourage a small subset of weights within the network to play the role of the entire network, enabling accurate classification of input samples. Essentially, these regularization terms, along with the corresponding parameter-sharing subnetworks, promote the sparsity of PrunNet, thereby enhancing its generalization ability. Consequently, dense networks optimized using our method exhibit superior performance on various benchmarks compared to those trained independently, as demonstrated by our experimental results.

E. Detailed experimental results

In this section, we present the detailed experimental results over the landmark benchmarks, including RParis [9], ROxford [9], and GLDv2-test [15].

Table A reports the performance of the dense network and subnetworks at pre-determined capacities. Our method outperforms the others in terms of both self-test and cross-test performance for the dense network and most subnetworks across these three datasets.

The detailed experimental results using different architectures are shown in Table B. Our method achieves the best performance over RParis, Roxford, and GLDv2-test on these representative architectures, indicating its strong generalization ability.

The detailed results of the experiments for simulating the deployment demand on new platforms are shown in Table C. For the methods without our PrunNet, we employ BCT [12] or SSPL [16] to train the subnetwork at 10% capacity compatible with the dense network, while for the methods with PrunNet, we conduct pruning by choosing the parameters with top-10% score. Our method achieves the best performance of the subnetwork at 10% capacity, demonstrating the effectiveness of our method and the flexibility for multi-platform deployments.

We also present detailed results of ablation studies on each landmark dataset in Table D. These detailed experimental results are consistent with the average results reported in the main manuscript, confirming the effectiveness of the proposed techniques.

F. Experiments on additional benchmarks

We carry out additional experiments on the following datasets to validate the generalization of our method: **(1) Market-1501 [20]**: A person re-identification dataset containing 32,668 images of 1,501 identities captured by 6 cameras. We use the standard split of 12,936 training images (751 identities) and 19,732 testing images (750 identities). **(2) MSMT17 [14]**: A large-scale person re-identification dataset with 126,441 images of 4,101 identities captured by 15 cameras. We adopt the split of 32,621 training images (1,041 identities) and 93,820 testing images (3,060 identities). **(3) CUB-200-2011 [13]**: A fine-grained bird classification dataset with 11,788 images of 200 bird

species. We use the standard split of 5,994 training images and 5,794 testing images.

The experimental results are presented in Table E, Table F and Table G, respectively. For Market-1501 and MSMT17 experiments, we employ ResNet-18 as the backbone while adopting ViT-S for CUB-200 experiments. Our method achieves state-of-the-art performance on both self-test and cross-test, validating the effectiveness and generalization of our proposed PrunNet. In particular, we found that CUB-200 with 5,994 training images is insufficient to train ViT-S from scratch. Hence, we pretrained all models on ImageNet-1K [1] before fine-tuning them on CUB-200.

G. Further exploration on structured pruning

Unlike structured pruning which preserves contiguous parameter blocks compatible with hardware computation units, unstructured pruning produces irregular sparse parameters, making it challenging to achieve actual acceleration on hardware implementations. To demonstrate the practical advantages of our method implemented by unstructured pruning, we present the storage usage (in the COO format) and theoretical FLOPs in Figure F.

We further conduct structured pruning experiments to explore a hardware-efficient method to generate compatible subnetworks. To achieve this, we implement a kernel-level score aggregation scheme, where pruning decisions are made by averaging importance scores within each convolutional kernel and removing kernels with the lowest aggregated scores. This approach enables PrunNet to directly leverage structured pruning mechanisms while maintaining architectural integrity. As presented in Table H, the structured pruning variant exhibits a moderate performance drop compared to the unstructured one, which is consistent with typical trends. Nevertheless, it outperforms SFSC, demonstrating its potential for structured sparsity. We will continue exploring structured PrunNet in future work.

H. Convergence analyses

In this section, we provide theoretical analyses of the convergence of our PrunNet and optimization algorithm.

H.1. Convergence analyses of greedy pruning

We analyze the convergence of greedy pruning in the following. According to the gradient calculated by Eq. (2) in the main manuscript, the update of score s_{ij}^l can be formulated as follows:

$$\tilde{s}_{ij}^l = s_{ij}^l - \eta \frac{\partial \mathcal{L}(\mathcal{I}_i^l)}{\partial \mathcal{I}_i^l} w_{ij}^l \mathcal{Z}_j^{l-1}. \tag{1}$$

If the connection (i, j) is replaced by (i, k) after the update, we can conclude that $s_{ij}^l > s_{ik}^l$ but $\tilde{s}_{ij}^l < \tilde{s}_{ik}^l$. Hence we have the following inequality:

$$\tilde{s}_{ij}^l - s_{ij}^l < \tilde{s}_{ik}^l - s_{ik}^l. \tag{2}$$

Based on Eq. (1), we can derive the inequality:

$$-\eta \frac{\partial \mathcal{L}(\mathcal{I}_i^l)}{\partial \mathcal{I}_i^l} w_{ij}^l \mathcal{Z}_j^{l-1} < -\eta \frac{\partial \mathcal{L}(\mathcal{I}_i^l)}{\partial \mathcal{I}_i^l} w_{ik}^l \mathcal{Z}_k^{l-1}. \tag{3}$$

We denote $\tilde{\mathcal{I}}_i^l$ as the new input to the i-th neuron at the l-th layer n_i^l after the replacement, and denote $\tilde{w}_{ik}^l$ as the new weight of the connection between n_i^l and n_k^{l-1}. Our goal is to prove the convergence of the loss, which can be formulated as $\mathcal{L}(\tilde{\mathcal{I}}_i^l) < \mathcal{L}(\mathcal{I}_i^l)$. According to Eq. (1) in the main manuscript, we have:

$$\tilde{\mathcal{I}}_i^l - \mathcal{I}_i^l = \tilde{w}_{ik}^l \mathcal{Z}_k^{l-1} - w_{ij}^l \mathcal{Z}_j^{l-1}. \tag{4}$$

Assuming the loss is smooth and $\tilde{\mathcal{I}}_i^l$ is close to $\mathcal{I}_i^l$, we can perform a Taylor expansion of the loss at $\mathcal{I}_i^l$ ignoring the second-order term, as shown in the follows:

$$\begin{aligned}
&\mathcal{L}(\tilde{\mathcal{I}}_i^l) = \mathcal{L}(\mathcal{I}_i^l + (\tilde{\mathcal{I}}_i^l - \mathcal{I}_i^l)) \\
&\leq \mathcal{L}(\mathcal{I}_i^l) + \frac{\partial \mathcal{L}(\mathcal{I}_i^l)}{\partial \mathcal{I}_i^l}(\tilde{\mathcal{I}}_i^l - \mathcal{I}_i^l) \\
&= \mathcal{L}(\mathcal{I}_i^l) + \frac{\partial \mathcal{L}(\mathcal{I}_i^l)}{\partial \mathcal{I}_i^l}(\tilde{w}_{ik}^l \mathcal{Z}_k^{l-1} - w_{ij}^l \mathcal{Z}_j^{l-1}) \\
&= \mathcal{L}(\mathcal{I}_i^l) + \frac{\partial \mathcal{L}(\mathcal{I}_i^l)}{\partial \mathcal{I}_i^l}((w_{ik}^l - \eta \frac{\partial \mathcal{L}}{\partial w_{ik}^l}) \mathcal{Z}_k^{l-1} - w_{ij}^l \mathcal{Z}_j^{l-1}) \\
&= \mathcal{L}(\mathcal{I}_i^l) + \frac{\partial \mathcal{L}(\mathcal{I}_i^l)}{\partial \mathcal{I}_i^l}(w_{ik}^l \mathcal{Z}_k^{l-1} - w_{ij}^l \mathcal{Z}_j^{l-1}) \\
&\quad - \eta \frac{\partial \mathcal{L}(\mathcal{I}_i^l)}{\partial \mathcal{I}_i^l} \frac{\partial \mathcal{L}}{\partial w_{ik}^l} \mathcal{Z}_k^{l-1} \\
&= \mathcal{L}(\mathcal{I}_i^l) + \frac{\partial \mathcal{L}(\mathcal{I}_i^l)}{\partial \mathcal{I}_i^l}(w_{ik}^l \mathcal{Z}_k^{l-1} - w_{ij}^l \mathcal{Z}_j^{l-1}) - \eta (\frac{\partial \mathcal{L}}{\partial w_{ik}^l})^2.
\end{aligned} \tag{5}$$

From Eq. (3), we have $\frac{\partial \mathcal{L}(\mathcal{I}_i^l)}{\partial \mathcal{I}_i^l}(w_{ik}^l \mathcal{Z}_k^{l-1} - w_{ij}^l \mathcal{Z}_j^{l-1}) < 0$. Thus we have proven that $\mathcal{L}(\tilde{\mathcal{I}}_i^l) < \mathcal{L}(\mathcal{I}_i^l)$, indicating the convergence of our greedy pruning scheme.

H.2. Convergence analyses of gradient integration

We analyze the convergence of the proposed conflict-aware gradient integration algorithm using a two-task learning example, where two losses $\mathcal{L}_1$ and $\mathcal{L}_2$ are optimized simultaneously. In this case, the network is optimized with the total loss $\mathcal{L} = \mathcal{L}_1 + \mathcal{L}_2$ where conflict-aware gradient integration is introduced to handle the gradient conflicting issue. We assume that $\mathcal{L}_1$ and $\mathcal{L}_2$ are convex and differentiable, and that the gradient of $\mathcal{L}$ is L-Lipschitz continuous with $L > 0$. A learning rate $\eta \leq \frac{1}{L}$ is used in the conflict-aware

gradient integration scheme to update the parameters. Our goal is to prove $\mathcal{L}(\tilde{\theta}) < \mathcal{L}(\theta)$, where θ is the parameters, $\tilde{\theta}$ is the new parameters updated with our conflict-aware gradient integration scheme.

Denoting the gradients of $\mathcal{L}_1$ and $\mathcal{L}_2$ by $\boldsymbol{g}_1$ and $\boldsymbol{g}_2$, respectively, if their cosine similarity $\langle \boldsymbol{g}_1, \boldsymbol{g}_2 \rangle \geq 0$, we directly calculate the summation of $\boldsymbol{g}_1$ and $\boldsymbol{g}_2$, which equals to the gradient of $\mathcal{L}$, to update the network. Given that $\eta \leq \frac{1}{L}$, the total loss $\mathcal{L}$ will decrease unless $\nabla\mathcal{L} = 0$ in this situation. Next we discuss the situation where $\langle \boldsymbol{g}_1, \boldsymbol{g}_2 \rangle < 0$. Assuming that $\nabla\mathcal{L}$ is L-Lipschitz continuous, we can conclude that $\nabla^2\mathcal{L}(\theta) - LI$ is a negative semi-definite matrix. We then conduct a quadratic expansion of $\mathcal{L}$ around $\mathcal{L}(\theta)$, which leads to the following inequality:

$$\begin{aligned}\mathcal{L}(\tilde{\theta}) &\leq \mathcal{L}(\theta) + \nabla\mathcal{L}(\theta)^T(\tilde{\theta} - \theta) + \frac{1}{2}\nabla^2\mathcal{L}(\theta) \parallel \tilde{\theta} - \theta \parallel^2 \\ &\leq \mathcal{L}(\theta) + \nabla\mathcal{L}(\theta)^T(\tilde{\theta} - \theta) + \frac{1}{2}L \parallel \tilde{\theta} - \theta \parallel^2 .\end{aligned} \tag{6}$$

Based on Eq. (3) in the main manuscript, we have:

$$\tilde{\theta} - \theta = -\eta\tilde{\boldsymbol{g}} = -n\eta(a\hat{\boldsymbol{g}}_1 + b\hat{\boldsymbol{g}}_2), \tag{7}$$

where $\hat{\boldsymbol{g}}_1$ and $\hat{\boldsymbol{g}}_2$ denote the gradient after projection, a and b denote the cosine similarity between $(\boldsymbol{g}_1, \hat{\boldsymbol{g}}_1)$ and $(\boldsymbol{g}_2, \hat{\boldsymbol{g}}_2)$, respectively. n represents the normalization coefficient, whose value equals $\frac{2}{a+b}$. Considering that $\nabla\mathcal{L}(\theta) = \boldsymbol{g} = \boldsymbol{g}_1 + \boldsymbol{g}_2$, Eq. (6) can be reformulated as:

$$\begin{aligned}\mathcal{L}(\tilde{\theta}) &\leq \mathcal{L}(\theta) - n\eta\boldsymbol{g}^T(a\hat{\boldsymbol{g}}_1 + b\hat{\boldsymbol{g}}_2) + \frac{1}{2}n^2L\eta^2 \parallel a\hat{\boldsymbol{g}}_1 + b\hat{\boldsymbol{g}}_2 \parallel^2 \\ &\leq \mathcal{L}(\theta) - n\eta\boldsymbol{g}^T(a\hat{\boldsymbol{g}}_1 + b\hat{\boldsymbol{g}}_2) + \frac{1}{2}n^2\eta \parallel a\hat{\boldsymbol{g}}_1 + b\hat{\boldsymbol{g}}_2 \parallel^2 \\ &= \mathcal{L}(\theta) - n\eta(\boldsymbol{g}_1 + \boldsymbol{g}_2)^T(a\hat{\boldsymbol{g}}_1 + b\hat{\boldsymbol{g}}_2) \\ &\quad + \frac{1}{2}n^2\eta(a^2 \parallel \hat{\boldsymbol{g}}_1 \parallel^2 + b^2 \parallel \hat{\boldsymbol{g}}_2 \parallel^2 + 2ab\hat{\boldsymbol{g}}_1 \cdot \hat{\boldsymbol{g}}_2) \\ &= \mathcal{L}(\theta) - n\eta(a\boldsymbol{g}_1 \cdot \hat{\boldsymbol{g}}_1 + b\boldsymbol{g}_2 \cdot \hat{\boldsymbol{g}}_2 + a\hat{\boldsymbol{g}}_1 \cdot \boldsymbol{g}_2 + b\hat{\boldsymbol{g}}_2 \cdot \boldsymbol{g}_1 \\ &\quad - \frac{1}{2}na^2 \parallel \hat{\boldsymbol{g}}_1 \parallel^2 - \frac{1}{2}nb^2 \parallel \hat{\boldsymbol{g}}_2 \parallel^2 - nab\hat{\boldsymbol{g}}_1 \cdot \hat{\boldsymbol{g}}_2).\end{aligned} \tag{8}$$

Given that $\hat{\boldsymbol{g}}_1 \cdot \boldsymbol{g}_2 = 0$ and $\hat{\boldsymbol{g}}_2 \cdot \boldsymbol{g}_1 = 0$, we can derive:

$$\begin{aligned}\mathcal{L}(\tilde{\theta}) \leq \mathcal{L}(\theta) &- n\eta(a\boldsymbol{g}_1 \cdot \hat{\boldsymbol{g}}_1 + b\boldsymbol{g}_2 \cdot \hat{\boldsymbol{g}}_2 \\ &- \frac{1}{2}na^2 \parallel \hat{\boldsymbol{g}}_1 \parallel^2 - \frac{1}{2}nb^2 \parallel \hat{\boldsymbol{g}}_2 \parallel^2 - nab\hat{\boldsymbol{g}}_1 \cdot \hat{\boldsymbol{g}}_2).\end{aligned} \tag{9}$$

Herein a and b are the cosine similarity between $(\boldsymbol{g}_1, \hat{\boldsymbol{g}}_1)$ and $(\boldsymbol{g}_2, \hat{\boldsymbol{g}}_2)$, respectively. We have

$$\begin{aligned}a\boldsymbol{g}_1 \cdot \hat{\boldsymbol{g}}_1 &= a^2 \parallel \boldsymbol{g}_1 \parallel \parallel \hat{\boldsymbol{g}}_1 \parallel = a \parallel \hat{\boldsymbol{g}}_1 \parallel^2, \\ b\boldsymbol{g}_2 \cdot \hat{\boldsymbol{g}}_2 &= b^2 \parallel \boldsymbol{g}_2 \parallel \parallel \hat{\boldsymbol{g}}_2 \parallel = b \parallel \hat{\boldsymbol{g}}_2 \parallel^2 .\end{aligned} \tag{10}$$

Then we get:

$$\begin{aligned}\mathcal{L}(\tilde{\theta}) &\leq \mathcal{L}(\theta) - n\eta(a \parallel \hat{\boldsymbol{g}}_1 \parallel^2 + b \parallel \hat{\boldsymbol{g}}_2 \parallel^2 \\ &\quad - \frac{1}{2}na^2 \parallel \hat{\boldsymbol{g}}_1 \parallel^2 - \frac{1}{2}nb^2 \parallel \hat{\boldsymbol{g}}_2 \parallel^2 - nab\hat{\boldsymbol{g}}_1 \cdot \hat{\boldsymbol{g}}_2) \\ &= \mathcal{L}(\theta) - n\eta((a - \frac{1}{2}na^2) \parallel \hat{\boldsymbol{g}}_1 \parallel^2 \\ &\quad + ((b - \frac{1}{2}nb^2) \parallel \hat{\boldsymbol{g}}_2 \parallel^2 - nab\hat{\boldsymbol{g}}_1 \cdot \hat{\boldsymbol{g}}_2) \\ &= \mathcal{L}(\theta) - n\eta\frac{ab}{a+b}(\parallel \hat{\boldsymbol{g}}_1 \parallel^2 + \parallel \hat{\boldsymbol{g}}_2 \parallel^2 - 2\hat{\boldsymbol{g}}_1 \cdot \hat{\boldsymbol{g}}_2)) \\ &= \mathcal{L}(\theta) - n\eta\frac{ab}{a+b}(\parallel \hat{\boldsymbol{g}}_1 - \hat{\boldsymbol{g}}_2 \parallel^2).\end{aligned} \tag{11}$$

Since the angle between the vectors before and after projection is less than $\frac{\pi}{2}$, we have $a, b \in (0, 1)$ and $\frac{ab}{a+b} > 0$. Thus, we have proven that $\mathcal{L}(\tilde{\theta}) < \mathcal{L}(\theta)$, indicating the convergence of our conflict-aware gradient integration scheme.

References

[1] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009. 8

[2] Alexey Dosovitskiy. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, pages 1–22, 2020. 1

[3] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016. 1

[4] Lingxiao He, Xingyu Liao, Wu Liu, Xinchen Liu, Peng Cheng, and Tao Mei. Fastreid: A pytorch toolbox for general instance re-identification. In *Proceedings of the 31st ACM International Conference on Multimedia*, pages 9664–9667, 2023. 1

[5] Bailin Li, Bowen Wu, Jiang Su, and Guangrun Wang. Eagleeye: Fast sub-net evaluation for efficient neural network pruning. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16*, pages 639–654. Springer, 2020. 1

[6] Xinchen Liu, Wu Liu, Tao Mei, and Huadong Ma. A deep learning-based approach to progressive vehicle re-identification for urban surveillance. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part II 14*, pages 869–884. Springer, 2016. 1

[7] Ziwei Liu, Ping Luo, Shi Qiu, Xiaogang Wang, and Xiaoou Tang. Deepfashion: Powering robust clothes recognition and retrieval with rich annotations. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1096–1104, 2016. 1

[8] Tan Pan, Furong Xu, Xudong Yang, Sifeng He, Chen Jiang, Qingpei Guo, Feng Qian, Xiaobo Zhang, Yuan Cheng, Lei

Yang, et al. Boundary-aware backward-compatible representation via adversarial learning in image retrieval. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15201–15210, 2023. 1

[9] Filip Radenović, Ahmet Iscen, Giorgos Tolias, Yannis Avrithis, and Ondřej Chum. Revisiting oxford and paris: Large-scale image retrieval benchmarking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5706–5715, 2018. 2, 3, 4, 5, 6, 7

[10] Vivek Ramanujan, Mitchell Wortsman, Aniruddha Kembhavi, Ali Farhadi, and Mohammad Rastegari. What's hidden in a randomly weighted neural network? In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11893–11902, 2020. 1

[11] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4510–4520, 2018. 1

[12] Yantao Shen, Yuanjun Xiong, Wei Xia, and Stefano Soatto. Towards backward-compatible representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6368–6377, 2020. 5, 7

[13] Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The caltech-ucsd birds-200-2011 dataset. 2011. 7

[14] Longhui Wei, Shiliang Zhang, Wen Gao, and Qi Tian. Person transfer gan to bridge domain gap for person re-identification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 79–88, 2018. 6, 7

[15] Tobias Weyand, Andre Araujo, Bingyi Cao, and Jack Sim. Google landmarks dataset v2-a large-scale benchmark for instance-level recognition and retrieval. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2575–2584, 2020. 1, 2, 3, 4, 5, 6, 7

[16] Hui Wu, Min Wang, Wengang Zhou, and Houqiang Li. Structure similarity preservation learning for asymmetric image retrieval. *IEEE Transactions on Multimedia*, pages 4693–4705, 2023. 5, 7

[17] Shengsen Wu, Yan Bai, Yihang Lou, Xiongkun Linghu, Jianzhong He, and Ling-Yu Duan. Switchable representation learning framework with self-compatibility. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15943–15953, 2023. 6

[18] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1492–1500, 2017. 1

[19] Andrew Zhai and Hao-Yu Wu. Classification is a strong baseline for deep metric learning. *arXiv preprint arXiv:1811.12649*, pages 1–12, 2018. 1

[20] Liang Zheng, Liyue Shen, Lu Tian, Shengjin Wang, Jingdong Wang, and Qi Tian. Scalable person re-identification: A benchmark. In *Proceedings of the IEEE international conference on computer vision*, pages 1116–1124, 2015. 6, 7