Supplementary Materials

A. More about Geometric Distillation: Proof and Simplicity

$$
\begin{aligned}
&\mathcal{L}_{\mathcal{G}}^{\boldsymbol{rt}}(\Delta\mathcal{Z}^{n-1},\Delta\mathcal{Z}^{n}) \\
&= \mathop{\mathbb{E}}_{\substack{(\hat{\boldsymbol{x}}_i,\hat{\boldsymbol{x}}_j,\hat{\boldsymbol{x}}_p,\hat{\boldsymbol{x}}_q)\in\mathbb{P};\\ \{i,j\}\neq\{p,q\}}}\Big[\mathcal{L}_{cos}(\Delta\boldsymbol{z}_{ij}^{n-1},\Delta\boldsymbol{z}_{ij}^{n})+\mathcal{L}_{cos}(\Delta\boldsymbol{z}_{pq}^{n-1},\Delta\boldsymbol{z}_{pq}^{n})+\mathcal{L}_{cos}(\Delta\boldsymbol{z}_{ij}^{n-1}-\Delta\boldsymbol{z}_{pq}^{n-1},\Delta\boldsymbol{z}_{ij}^{n}-\Delta\boldsymbol{z}_{pq}^{n})\Big] \\
&= \underbrace{\lambda_1\mathop{\mathbb{E}}_{\substack{(\hat{\boldsymbol{x}}_i,\hat{\boldsymbol{x}}_j,\hat{\boldsymbol{x}}_p)\in\mathbb{P};\\ i\neq j\neq p}}\Big[\mathcal{L}_{cos}(\Delta\boldsymbol{z}_{ij}^{n-1},\Delta\boldsymbol{z}_{ij}^{n})+\mathcal{L}_{cos}(\Delta\boldsymbol{z}_{ip}^{n-1},\Delta\boldsymbol{z}_{ip}^{n})+\mathcal{L}_{cos}(\Delta\boldsymbol{z}_{ij}^{n-1}-\Delta\boldsymbol{z}_{ip}^{n-1},\Delta\boldsymbol{z}_{ij}^{n}-\Delta\boldsymbol{z}_{ip}^{n})\Big]}_{\text{triplet}} \\
&\quad+\underbrace{\lambda_2\mathop{\mathbb{E}}_{\substack{(\hat{\boldsymbol{x}}_i,\hat{\boldsymbol{x}}_j,\hat{\boldsymbol{x}}_p,\hat{\boldsymbol{x}}_q)\in\mathbb{P};\\ i\neq j\neq p\neq q}}\Big[\mathcal{L}_{cos}(\Delta\boldsymbol{z}_{ij}^{n-1},\Delta\boldsymbol{z}_{ij}^{n})+\mathcal{L}_{cos}(\Delta\boldsymbol{z}_{pq}^{n-1},\Delta\boldsymbol{z}_{pq}^{n})+\mathcal{L}_{cos}(\Delta\boldsymbol{z}_{ij}^{n-1}-\Delta\boldsymbol{z}_{pq}^{n-1},\Delta\boldsymbol{z}_{ij}^{n}-\Delta\boldsymbol{z}_{pq}^{n})\Big]}_{\text{quadruplet}} \\
&= \lambda_1\mathop{\mathbb{E}}_{\substack{(\hat{\boldsymbol{x}}_i,\hat{\boldsymbol{x}}_j,\hat{\boldsymbol{x}}_p)\in\mathbb{P};\\ i\neq j\neq p}}\Big[\mathcal{L}_{cos}(\Delta\boldsymbol{z}_{ij}^{n-1},\Delta\boldsymbol{z}_{ij}^{n})+\mathcal{L}_{cos}(\Delta\boldsymbol{z}_{ip}^{n-1},\Delta\boldsymbol{z}_{ip}^{n})+\mathcal{L}_{cos}(\Delta\boldsymbol{z}_{pj}^{n-1},\Delta\boldsymbol{z}_{pj}^{n})\Big] \\
&\quad+\lambda_2\mathop{\mathbb{E}}_{\substack{(\hat{\boldsymbol{x}}_i,\hat{\boldsymbol{x}}_j,\hat{\boldsymbol{x}}_p,\hat{\boldsymbol{x}}_q)\in\mathbb{P};\\ i\neq j\neq p\neq q}}\Big[\mathcal{L}_{cos}(\Delta\boldsymbol{z}_{ij}^{n-1},\Delta\boldsymbol{z}_{ij}^{n})+\mathcal{L}_{cos}(\Delta\boldsymbol{z}_{pq}^{n-1},\Delta\boldsymbol{z}_{pq}^{n})+\mathcal{L}_{cos}(\Delta\boldsymbol{z}_{ij}^{n-1}-\Delta\boldsymbol{z}_{pq}^{n-1},\Delta\boldsymbol{z}_{ij}^{n}-\Delta\boldsymbol{z}_{pq}^{n})\Big] \\
&= \underbrace{\lambda_3\mathop{\mathbb{E}}_{\substack{(\hat{\boldsymbol{x}}_i,\hat{\boldsymbol{x}}_j)\in\mathbb{P};\\ i\neq j}}\Big[\mathcal{L}_{cos}(\Delta\boldsymbol{z}_{ij}^{n-1},\Delta\boldsymbol{z}_{ij}^{n})\Big]}_{\Delta\text{ part}}+\underbrace{\lambda_4\mathop{\mathbb{E}}_{\substack{(\hat{\boldsymbol{x}}_i,\hat{\boldsymbol{x}}_j,\hat{\boldsymbol{x}}_p,\hat{\boldsymbol{x}}_q)\in\mathbb{P}\\ i\neq j\neq p\neq q}}\Big[\mathcal{L}_{cos}(\Delta\boldsymbol{z}_{ij}^{n-1}-\Delta\boldsymbol{z}_{pq}^{n-1},\Delta\boldsymbol{z}_{ij}^{n}-\Delta\boldsymbol{z}_{pq}^{n})\Big]}_{\text{residual }\Delta\text{ part}},
\end{aligned}
\tag{1}
$$

where λ_1, λ_2, λ_3 and λ_4 are constants with respect to weights of corresponding parts in the batch, $(\hat{\boldsymbol{x}}_i,\hat{\boldsymbol{x}}_j,...)\in\mathbb{P}$ means these samples share the same ID. When $\mathcal{L}_{\mathcal{G}}^{\boldsymbol{rt}}$ is optimal, Δ part should be the optimal that for any positive pair $(\hat{\boldsymbol{x}}_i,\hat{\boldsymbol{x}}_j)\in\mathbb{P}$, their residual feature vector in preceding feature space $\Delta\boldsymbol{z}_{ij}^{n-1}=\boldsymbol{z}_i^{n-1}-\boldsymbol{z}_j^{n-1}$ and in evolving feature space $\Delta\boldsymbol{z}_{ij}^{n}=\boldsymbol{z}_i^{n}-\boldsymbol{z}_j^{n}$ is parallel and with the same orientation. Then, for instance (Fig. 1), due to the fundamental AAA criterion of triangle similarity, $\triangle ABC\sim\triangle A'B'C'$. The scaling coefficient is $\frac{AB}{A'B'}=\frac{AC}{A'C'}=\frac{BC}{B'C'}=r$ and translation vector is $\boldsymbol{t}$, *i.e.*, $\overrightarrow{OA}+\boldsymbol{t}=\overrightarrow{OA'}$, where O is the Origin (not marked in Fig.). Because the sides are parallel with the same orientations and the triangles are arbitrary, no rotation and reflection is allowed. As the adjacent triangles, $\triangle ABD\sim\triangle A'B'D'$ and share the same r and $\boldsymbol{t}$, *i.e.*, $\frac{AD}{A'D'}=\frac{AB}{A'B'}=\frac{BD}{B'D'}=r,\overrightarrow{OD}+\boldsymbol{t}=\overrightarrow{OD'}$. Similarly, all triangles in polyhedron share the same r and $\boldsymbol{t}$, which implies the polyhedron in evolving space is similar to that in preceding space with scaling coefficient r and translation vector $\boldsymbol{t}$. Then we can simplify $\mathcal{L}_{\mathcal{G}}^{\boldsymbol{rt}}$ as:

$$
\mathcal{L}_{\mathcal{G}}^{\boldsymbol{rt}}(\Delta\mathcal{Z}^{n-1},\Delta\mathcal{Z}^{n})=\mathop{\mathbb{E}}_{\substack{(\hat{\boldsymbol{x}}_i,\hat{\boldsymbol{x}}_j)\in\mathbb{P};\\ i\neq j}}\Big[\mathcal{L}_{cos}(\Delta z_{ij}^{n-1},\Delta z_{ij}^{n})\Big].
\tag{2}
$$

Note that the residual Δ part will be optimal when the Δ part is optimal, so omitting the residual Δ part will not change the objective. All AGD / GD experiments in our work are based on Equ. 2 unless otherwise specified.

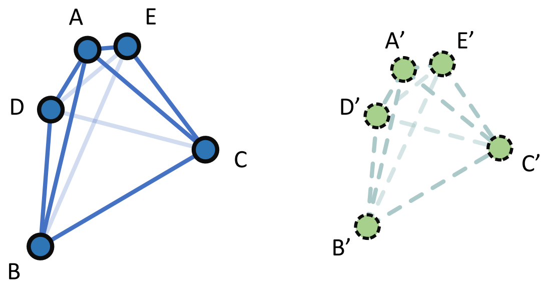

Figure 1. Illustration for proof. The polyhedron in preceding feature space (blue) and the polyhedron in evolving feature space (green) are encouraged to be geometrically similar.

B. More about Augmented Distillation

In the default setting, our augmented distillation has 2 peers, that one dreaming image is augmented into 2 views. When *peers* is greater than 2, we rewrite the $\mathcal{L}_{\mathcal{AD}}$ as below:

$$\begin{aligned}&\mathcal{L}^{\mathcal{P}}_{\mathcal{AD}}(\hat{\boldsymbol{x}}^s, \mathcal{P}; \mathcal{L}_\kappa)\\ &\quad= \alpha\mathcal{L}_\kappa(\hat{\boldsymbol{x}}^s, \hat{\boldsymbol{x}}^s) + \sum_{\substack{\hat{\boldsymbol{x}}^t\in\mathcal{P}\\ \hat{\boldsymbol{x}}^t\neq\hat{\boldsymbol{x}}^s}} \frac{1-\alpha}{|\mathcal{P}|-1}\mathcal{L}_\kappa(\hat{\boldsymbol{x}}^s, \hat{\boldsymbol{x}}^t),\end{aligned} \tag{3}$$

$$\mathcal{L}_{\mathcal{AD}}(\mathcal{P}; \mathcal{L}_\kappa) = \frac{1}{|\mathcal{P}|}\sum_{\hat{\boldsymbol{x}}^s\in\mathcal{P}} \mathcal{L}^{\mathcal{P}}_{\mathcal{AD}}(\hat{\boldsymbol{x}}^s, \mathcal{P}; \mathcal{L}_\kappa), \tag{4}$$

where $\mathcal{P}$ is the set of peers, *i.e.*, $\mathcal{P} = \{\hat{\boldsymbol{x}}', \hat{\boldsymbol{x}}'', \hat{\boldsymbol{x}}''', ...\}$, and *peers* $= |\mathcal{P}|$. In order to reduce the GRAM when adopting AD mechanism, we divide $\mathcal{L}_{\mathcal{AD}}$ into $\mathcal{L}^{\mathcal{P}}_{\mathcal{AD}}$s and reformulate the algorithm as below:

Algorithm 1 Augmented Geometric Distillation (*n*-th task)

```
Input: Incremental dataset $\mathcal{D}_{T_n}$ and fixed base model $f^{n-1}_{\boldsymbol{\theta}}$.
Output: Converged evolving model $f^n_{\boldsymbol{\theta}}$.
1:  Generate dreaming memory $\mathcal{M}_{T_{1:n-1}}$ with $f^{n-1}_{\boldsymbol{\theta}}$.
2:  Initialize the evolving model $f^n_{\boldsymbol{\theta}}$ with $f^{n-1}_{\boldsymbol{\theta}}$.
3:  while not converged do
4:      Sample $\boldsymbol{x} \subset \mathcal{D}_{T_n}$.
5:      Sample and augment twice $\hat{\boldsymbol{x}} \subset \mathcal{M}_{T_{1:n-1}} \rightarrow \mathcal{P}$.
6:      for $\hat{\boldsymbol{x}}^s \in \mathcal{P}$ do
7:          Augment $\boldsymbol{x} \rightarrow \boldsymbol{x}'$. // random aug each time.
8:          Calculate $\mathcal{L}_{rep}$ with $f^n_{\boldsymbol{\theta}}(\boldsymbol{x}')$ and $f^n_{\boldsymbol{\theta}}(\hat{\boldsymbol{x}}^s)$.
9:          Calculate $\mathcal{L}^{\mathcal{P}}_{\mathcal{AD}}(\hat{\boldsymbol{x}}^s, \mathcal{P}; \mathcal{L}^{rt}_{\mathcal{G}})$ between $\{f^{n-1}_{\boldsymbol{\theta}}(\hat{\boldsymbol{x}}^t)\}_{\hat{\boldsymbol{x}}^t\in\mathcal{P}}$
            and $f^n_{\boldsymbol{\theta}}(\hat{\boldsymbol{x}}^s)$.
10:         Calculate $\frac{1}{|\mathcal{P}|}\left[\mathcal{L}_{rep} + \lambda\mathcal{L}^{\mathcal{P}}_{\mathcal{AD}}(\hat{\boldsymbol{x}}^s, \mathcal{P}; \mathcal{L}^{rt}_{\mathcal{G}})\right]$ and back-
            ward.
11:     end for
12:     Update $\boldsymbol{\theta}$ in $f^n_{\boldsymbol{\theta}}$.
13: end while
14: Fix the evolving model $f^n_{\boldsymbol{\theta}}$ for the next step as base model.
```

Note that, when adopting AD, we augment incremental data $\boldsymbol{x}$ into *peers* views. The trick is mainly designed to correct the bias in BN layers. If much more dreaming data is fed into model, it will cause bias to the preceding data distribution in BN layers. Such bias is detrimental to ReID, which is sensitive to bias in BN as discussed in DualNorm [4]. And this trick brings no extra performance gain in supervised training.

Method	5 steps	10 steps	Method	5 steps	10 steps
LUCIR w/ $\mathcal{L}^{rt}_{\mathcal{G}}$	65.3	64.1	TPCIL	65.3	63.6
POD w/ AGD	65.7	65.3	POD w/ DDE	65.4	64.1

Table 1. Comparison on CIFAR100. Note that we uses NME as classifier for AGD and report results from original paper from TPCIL and DDE. TOPIC is not included due to its few-shot setting.

C. Further Discussion

C.1. Drawing Parallels with Other Topology / Geometric Works

Basically, other topological/geometric methods managed to preserve the cosine similarity, Mahalanobis distance, Euc distance or logits between neighbors to maintain topology. These methods only modulate relationship as a distance metric (which is a value) between samples, while GD modulates relationship as a vector in high-dimensional space. Therefor, richer information in channel-wise could also be preserved and GD constrains the geometric structure more strictly. More comparison on CIFAR100 is shown in Tab. 1.

C.2. Knowledge Transfer and Trade-off

Method	Data	MSMT17	
		mAP	Rank-1
Oracle	Real	46.2	71.6
Knowledge Transfer	Dreaming	45.2	70.7

Table 2. Extensive knowledge transfer experiments. Student model is trained with $\mathcal{L}_{\mathbf{AGD}}$ only on the dreaming data.

To provide deeper insight into the trade-off between learning and reviewing, we conduct knowledge transferring that $\mathcal{I}$ is fed into the student to distill knowledge from "Oracle" teacher model. The student model is initialized with ImageNet [10] pre-trained parameters. After convergence, the student is almost capable to act as well as teacher does (as illustrated in Tab. 2). However, when feeding the incremental data from another domain, the student model fails to reproduce the results in Tab. 2, which reveals that natural divergence exists between incremental data and $\mathcal{I}$ and further explains the contradiction between stabilizing the feature space for preceding domains and adapting feature space for the incremental domain. And also such trade-off between preceding and incremental task is still an open-issue in CIL and could be observed in all typical methods in CIL. When compared with these methods, AGD achieves better balance and overall performance.

C.3. Discussion with CBN

Testing Set		MSMT17	
Seq Length	Method	mAP	Rank-1
1	-	100%	100%
2	CBN (w/o exemplars)	58.9%	74.5%
	CBN (w/ exemplars)	84.6%	91.6%
	AGD	90.7%	94.6%
3	CBN (w/o exemplars)	39.4%	56.6%
	CBN (w/ exemplars)	76.0%	86.4%
	AGD	76.5%	85.5%

Table 3. Results on the percentages of preserved performance when MSMT17 acts as the first base task.

CBN [14] proposed a novel BN layer for ReID that input image is normalized with regard to its camera ID. CBN reports its significant improvements in incremental scenario. According to the protocol in CBN, we report the percentage of preserved performance after incremental learning when compared with "Oracle" (detailed in CBN Sec. 4.2). The results in Table 3 are the average preserved performance when MSMT17 serves as T_1 and leads task sequences with length of 2 or 3. The great improvements over CBN (w/o exemplars) demonstrate the superiority of our AGD framework and we believe this is because of dreaming memory for replaying and effective distillation. Note that AGD does not store exemplars, all memory is recovered by dreaming. And when compared with CBN (w/ exemplars), AGD still surpasses CBN in task sequences with length of 2 and is competitive in task sequences with length of 3.

D. More Implementation Details

D.1. More about $\mathcal{L}_{rep}$

In the baseline solution and our AGD framework, basic representation training is indispensable. Concretely, $\mathcal{L}_{rep}$ contains $\mathcal{L}_{ce}$ and $\mathcal{L}_{tri}$. Let networks $f^n_{\boldsymbol{\theta}}(\cdot)$ encode the input $(\boldsymbol{x}, y)$. Then the loss for representation is formulated as follow:

$$\mathcal{L}_{ce}(\boldsymbol{x}) = -\sum_i \delta_{y_i=y} \log[p(y_i|\boldsymbol{x}, f^n_{\boldsymbol{\theta}})], \tag{5}$$

$$\mathcal{L}_{tri}(\boldsymbol{x}) = softplus(\max d_{ap} - \min d_{an}), \tag{6}$$

where $p(y_i|\boldsymbol{x}, f^n_{\boldsymbol{\theta}})$ is the probability that $\boldsymbol{x}$ is an image of ID y_i. $\delta_{y_i=y}$ denotes the identity function that outputs 1 if $y_i = y$ and 0 otherwise. $softplus(\cdot)$ is expressed as $\log(1+\exp(\cdot))$. d_{ap} denotes Euclidean distances of positive pairs in mini-batches and d_{an} for negative. We simply add $\mathcal{L}_{ce}$ and $\mathcal{L}_{tri}$ as the loss for representation training:

$$\mathcal{L}_{rep}(\boldsymbol{x}) = \mathcal{L}_{ce}(\boldsymbol{x}) + \mathcal{L}_{tri}(\boldsymbol{x}). \tag{7}$$

D.2. More about $\mathcal{L}_{cos}, \mathcal{L}_{\ell 1}$ and $\mathcal{L}_{\ell 2}$

In our ablation studies in Sec. 5.4, we attempt to make comparisons between $\mathcal{L}_{\Delta}$ and $\mathcal{L}_{cos}, \mathcal{L}_{\ell 1}, \mathcal{L}_{\ell 2}$. $\mathcal{L}_{cos}$ expects embeddings of a certain exemplar output by student and teacher to have the same orientation, while $\mathcal{L}_{\ell 1}$ and $\mathcal{L}_{\ell 2}$ enforce the embeddings to remain static. We formulate them in detail as below:

$$\mathcal{L}_{cos}(\boldsymbol{z}^{n-1}, \boldsymbol{z}^n) = 1 - \langle \boldsymbol{z}^{n-1}, \boldsymbol{z}^n \rangle = 1 - \bar{\boldsymbol{z}}^{n-1} \cdot (\bar{\boldsymbol{z}}^n)^T, \tag{8}$$

$$\mathcal{L}_{\ell 1}(\boldsymbol{z}^{n-1}, \boldsymbol{z}^n) = \sum_i |z_i^{n-1} - z_i^n|, \tag{9}$$

$$\mathcal{L}_{\ell 2}(\boldsymbol{z}^{n-1}, \boldsymbol{z}^n) = \|\boldsymbol{z}^{n-1} - \boldsymbol{z}^n\|_2^2, \tag{10}$$

where $\bar{z}$ denotes the normalized feature $\boldsymbol{z}$, $\boldsymbol{z}_i$ is the i-th element in feature $\boldsymbol{z}$ and $\|\cdot\|_2$ stands for ℓ_2 distance. Note that $\mathcal{L}_{\ell 2}$ here is the MSE loss, where is equivalent to square of ℓ_2 distance.

D.3. More about Dreaming Memory

Concretely, a batch of images $\hat{\boldsymbol{x}}$, initialized with noise, and their random IDs $\hat{y}$ are fed into the fixed networks. To attach discriminative information in semantics level to the images, images are updated by gradients of cross-entropy loss from the networks. To further constrain the synthesized images located in base domains, images are expected to match the statistics in fixed BN layers with ℓ_2 loss.

The overall loss for synthesis is formulated as:

$$\begin{aligned}\mathcal{L}_{syn}(\hat{\boldsymbol{x}}) = \mathcal{L}_{ce}(\hat{\boldsymbol{x}}) + \alpha \sum_{l \in L} \|mean(\hat{\boldsymbol{x}}_l) - mean_l\|_2^2 \\ + \beta \sum_{l \in L} \|var(\hat{\boldsymbol{x}}_l) - var_l\|_2^2 + \mathcal{L}_{prior},\end{aligned} \tag{11}$$

where $\mathcal{L}_{ce}(\hat{\boldsymbol{x}})$ computes cross-entropy loss of $\hat{\boldsymbol{x}}$ as formulated in Equ.5. $mean(\hat{\boldsymbol{x}}_l)$ and $var(\hat{\boldsymbol{x}}_l)$ denote mean and variance vectors calculated on-the-fly in l-th BN layer and as their targets, $mean_l$ and var_l are mean and var vector recorded in l-th BN layer. $\mathcal{L}_{prior}$ constrains smoothness in pixel level, as detailed in DeepInversion [13].

After the generation, the dreaming stage builds a dreaming memory $\mathcal{M} = \{(\hat{\boldsymbol{x}}_i, \hat{y}_i)\}$ in base domains and each image $\hat{\boldsymbol{x}}$ is attached with discriminative feature of a certain ID $\hat{y}$.

Demonstration. As illustrated in Fig.2, synthesized images basically contain most discriminative regions of each ID. For example, in the second grid, a woman in khaki with a stripe handbag can be observed in both images. However, due to the generating strategy with gradients, the dreaming memory fails to recover details in original images and just estimate distributions of preceding data, which protects the privacy.

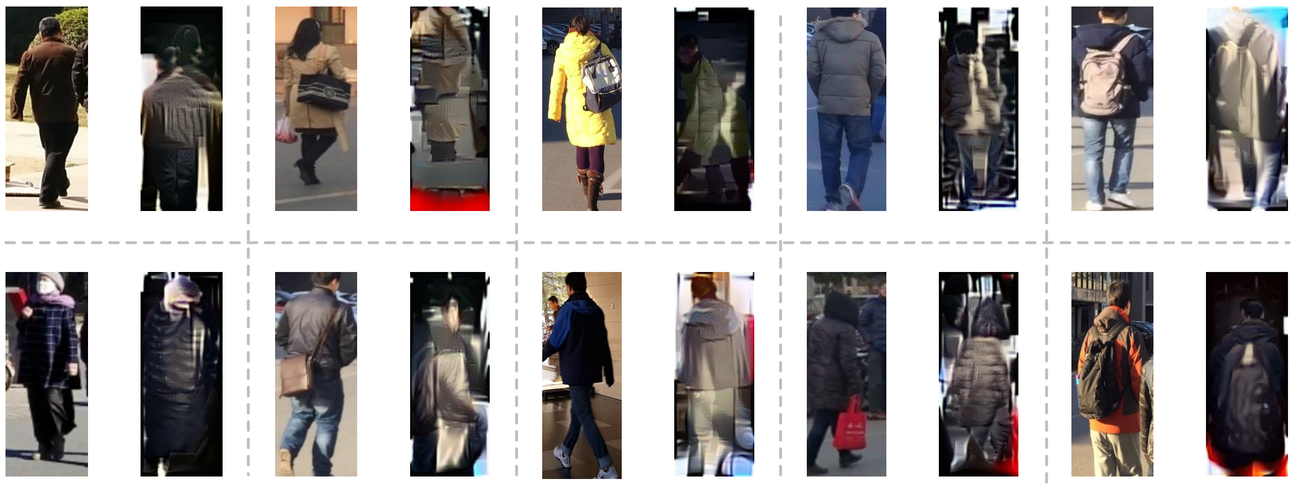

Figure 2. Comparison between synthesized images and raw images in MSMT17. In each grid, the raw image is shown on the left and corresponding synthesized image is shown on the right. IDs of these pairs, in sequence, are 118, 183, 377, 425, 32, 428, 615, 786, 1007 and 843.

Implementation Details. To generate the dreaming memory, we follow the training strategy in DeepInversion [13]. We set batch size to 64 and initialize each batch with Gaussian noise. Adam optimizer with learning rate 0.2 is adopted to update images. α and β is set to 0.01. $\mathcal{L}_{prior}$ consists of total variant ℓ_2 loss and ℓ_2 loss which is directly cast on the images. And their coefficients are 1e-4 and 1e-5 correspondingly. Before fed into the fixed networks, the inputs are augmented by random flipping. The two-stage accelerating is leveraged, as details in DeepInversion [13] (Sec. 4.4 b) Multi-resolution synthesis).

D.4. Details in Other Methods

λ denotes the weight of proposed loss or objectives for distillation or consolidation.

EWC [5]. $\lambda = 1e4$. The optimizer is Adam with initial learning rate of 1e-5.

MAS [1]. $\lambda = 1e5$. The optimizer is Adam with initial learning rate of 1e-5.

LwF [6]. $\lambda = 3, T = 2$, where T is the temperature in prediction. The optimizer is SGD with initial learning rate of 1e-2.

AKA [8]. We reproduce the experiments with official code here[1]. Note that we ran all experiments under our settings for fair comparisons (e.g. 1. joint loss of CE and triplet loss. 2. strong augmentation (REA `sh=0.4`). 3. many well-recognized tricks in BoT [7]). Our baseline settings lead to stronger supervision baseline and more forgetting than original results in AKA, or in short, our settings are more challenging.

iCaRL [9]. $\lambda = 2, T = 2$, where T is the temperature in prediction. The optimizer is SGD with initial learning rate of 1e-2.

ABD [12]. $\lambda = 2$. The optimizer is SGD with initial learning rate of 1e-2.

LUCIR (w/ cos) [3]. $\lambda = 5$. The optimizer is SGD with initial learning rate of 1e-2.

LUCIR (w/ ℓ_1) [3]. $\lambda = 0.005$. The optimizer is SGD with initial learning rate of 1e-2.

LUCIR (w/ ℓ_2) [3]. $\lambda = 0.005$. The optimizer is SGD with initial learning rate of 1e-2.

PODNet (w/ ℓ_2) [2]. $\lambda_1 = 5, \lambda_2 = 8$, where λ_1 is the weight for *cos* distillation term of final embeddings and λ_2 is the weight for *cos* distillation term of pooled intermediate attention. The optimizer is SGD with initial learning rate of 1e-2. We reproduce the distillation term with official code here[2].

GeoDL (w/ ℓ_2) [11]. $\lambda_1 = 5, \lambda_2 = 4$, where λ_1 is the weight for *cos* distillation term and λ_2 is the weight for GeoDL. The optimizer is SGD with initial learning rate of 1e-2. We reproduce the distillation term with official code here[3].

D.5. More about CIL

Nearest-Mean-of-Exemplars (NME). In CIL, the memory contains exemplars, which are the real images of preceding classes. Given fixed networks $f_{\boldsymbol{\theta}}$, we extract prototype vectors of all classes, *i.e.*, $\mu_{y_i} = \frac{1}{|P_{y_i}|}\sum_{x \in P_{y_i}} f_{\boldsymbol{\theta}}(x)$, where P_{y_i} is the exemplar set of class y_i. To predict the label of an image $\boldsymbol{x}$, we calculate the distance between its feature and all prototype vectors and

[1]https://github.com/TPCD/LifelongReID

[2]https://github.com/arthurdouillard/incremental_learning.pytorch

[3]https://github.com/chrysts/geodesic_continual_learning

assign the label, whose prototype vector is the most similar:

$$y^* = \underset{i}{\operatorname{argmin}} \|\mu_{y_i} - f_{\boldsymbol{\theta}}(\boldsymbol{x})\|_2. \tag{12}$$

We leverage NME not fully connected layer as our classifier mainly because it is similar to the retrieval process that ranking distances between image features and gallery in the feature space.

About How to Load Memory Data. Due to our geometric distillation, we have to load memory data with CK sampler, which loads C class and K samples for each class ($C \times K$ samples in total). To fullfil such request, we split batch-size into two parts (incremental data and memory data). For incremental data, we first calculate the percentage of incremental data in all data and assign the same part in each batch-size. For memory data, it takes and fills the rest part in each batch-size with $C \times K$ memory samples. And typically, $K = 4$.

Average Incremental Accuracy. After each incremental step, the model is tested on the test set of all seen classes and reports the accuracy. We average all these accuracies to get *average incremental accuracy* and report the final results after 3 runs with different random seeds.

References

[1] Rahaf Aljundi, Francesca Babiloni, Mohamed Elhoseiny, Marcus Rohrbach, and Tinne Tuytelaars. Memory aware synapses: Learning what (not) to forget. In *ECCV*, pages 139–154, 2018. 4

[2] Arthur Douillard, Matthieu Cord, Charles Ollion, Thomas Robert, and Eduardo Valle. Podnet: Pooled outputs distillation for small-tasks incremental learning. In *ECCV*, pages 86–102. Springer, 2020. 4

[3] Saihui Hou, Xinyu Pan, Chen Change Loy, Zilei Wang, and Dahua Lin. Learning a unified classifier incrementally via rebalancing. In *CVPR*, pages 831–839, 2019. 4

[4] Jieru Jia, Qiuqi Ruan, and Timothy Hospedales. Frustratingly easy person re-identification: Generalizing person reid in practice. In Kirill Sidorov and Yulia Hicks, editors, *BMVC*, pages 141.1–141.14. BMVA Press, September 2019. 2

[5] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *PNAS*, 114(13):3521–3526, 2017. 4

[6] Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE TPAMI*, 40(12):2935–2947, 2017. 4

[7] Hao Luo, Wei Jiang, Youzhi Gu, Fuxu Liu, Xingyu Liao, Shenqi Lai, and Jianyang Gu. A strong baseline and batch normalization neck for deep person re-identification. *TMM*, 2019. 4

[8] Nan Pu, Wei Chen, Yu Liu, Erwin M Bakker, and Michael S Lew. Lifelong person re-identification via adaptive knowledge accumulation. In *CVPR*, pages 7901–7910, 2021. 4

[9] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. icarl: Incremental classifier and representation learning. In *CVPR*, pages 2001–2010, 2017. 4

[10] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *IJCV*, 115(3):211–252, 2015. 2

[11] Christian Simon, Piotr Koniusz, and Mehrtash Harandi. On learning the geodesic path for incremental learning. In *CVPR*, pages 1591–1600, 2021. 4

[12] James Smith, Yen-Chang Hsu, Jonathan Balloch, Yilin Shen, Hongxia Jin, and Zsolt Kira. Always be dreaming: A new approach for data-free class-incremental learning. In *ICCV*, pages 9374–9384, October 2021. 4

[13] Hongxu Yin, Pavlo Molchanov, Jose M Alvarez, Zhizhong Li, Arun Mallya, Derek Hoiem, Niraj K Jha, and Jan Kautz. Dreaming to distill: Data-free knowledge transfer via deepinversion. In *CVPR*, pages 8715–8724, 2020. 3, 4

[14] Zijie Zhuang, Longhui Wei, Lingxi Xie, Tianyu Zhang, Hengheng Zhang, Haozhe Wu, Haizhou Ai, and Qi Tian. Rethinking the distribution gap of person re-identification with camera-based batch normalization. In *ECCV*, pages 140–157. Springer, 2020. 3