Mimicking the Oracle: An Initial Phase Decorrelation Approach for Class Incremental Learning (Appendix)

Yujun Shi[1] Kuangqi Zhou[1] Jian Liang[3] Zihang Jiang[1]
Jiashi Feng[2] Philip Torr[4] Song Bai[2] Vincent Y. F. Tan[1]
[1]National University of Singapore [2] ByteDance Inc.
[3] Institute of Automation, Chinese Academy of Sciences (CAS) [4] University of Oxford
shi.yujun@u.nus.edu vtan@nus.edu.sg

A. Proof of Proposition 1 (Main text Sec. 2.3)

Proof. Recall that for a d-by-d correlation matrix K, we have:

$$\sum_{i=1}^{d} \lambda_i^{(c)} = \mathrm{Tr}(K) = d. \tag{1}$$

This is because for a square matrix, summation of eigenvalues equals trace of the matrix. In addition, for a correlation matrix, all its diagonal elements are 1, which results in $\mathrm{Tr}(K) = d$.

Next, for the left hand side of the equation, we have:

$$\begin{aligned}\sum_{i=1}^{d}(\lambda_i - \frac{1}{d}\sum_{j=1}^{d}\lambda_j)^2 &= \sum_{i=1}^{d}(\lambda_i - 1)^2 \qquad \text{(Plug-in Eqn. (1))}\\ &= \sum_{i=1}^{d}\lambda_i^2 - 2\sum_{i=1}^{d}\lambda_i + d \\ &= \sum_{i=1}^{d}\lambda_i^2 - d \qquad \text{(Plug-in Eqn. (1)).}\end{aligned} \tag{2}$$

Next, for the right hand side, we have:

$$\begin{aligned}\|K\|_{\mathrm{F}}^2 - d &= \mathrm{Tr}(K^T K) - d \\ &= \mathrm{Tr}(U\Sigma U^T U\Sigma U^T) - d \quad \text{(Applying eigendecomposition on } K) \\ &= \mathrm{Tr}(U\Sigma^2 U^T) - d \\ &= \sum_{i=1}^{n}\lambda_i^2 - d.\end{aligned} \tag{3}$$

Therefore, we have shown that left hand side of the equation equals the right hand side. □

B. More Analysis on Why CwD Improves CIL

Although we have developed our method through observations on the differences between the oracle model and the naïvely-trained initial-phase model, the underlying reason why more uniformly scattered representations for each class benefit CIL is still yet to be explored.

In this section, we provide additional analysis on the effectiveness of CwD. We posit the reason why CwD improves CIL is as follow: after applying CwD, data representations produced by the model are not overly compressed. That being said,

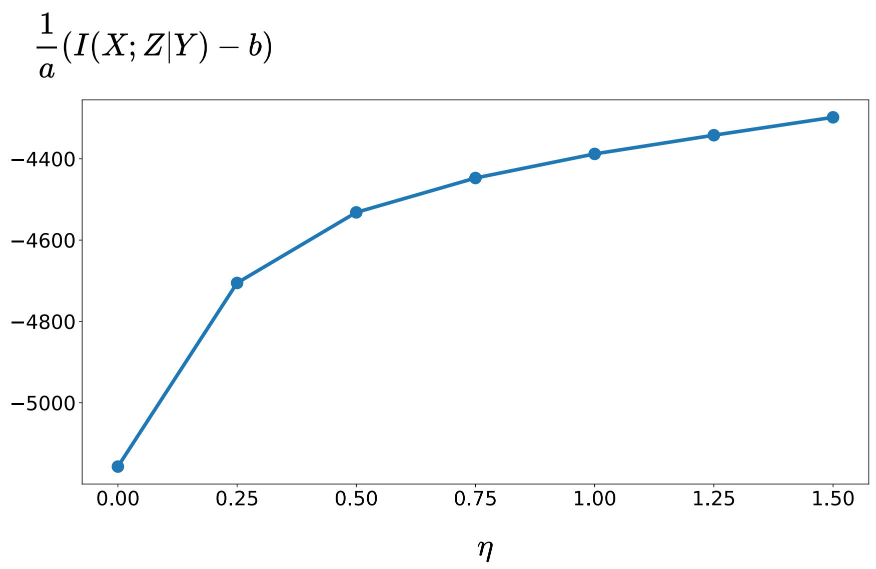

Figure 1. **Visualization on how $I(X;Z|Y)$ change with increasing η.** X-Axis is η. Y-Axis is $\frac{1}{C}\sum_{c=1}^{C}\sum_{i=1}^{d}\log\lambda_i^{(c)}=\frac{1}{a}(I(X;Z|Y)-b)$ as derived in Eqn. (5). Note that here Y-Axis is negative because it is conditional mutual information minus a very large positive constant.

the representations will contain more information about input data besides the information useful for classification at the initial phase. Although these additional information are not useful for classification at the initial phase, they can be useful for discriminating between classes of the initial phase and future phases, which will improve CIL when incrementally learning new classes.

To quantitatively validate that CwD helps preserve more information in representations beyond classification at initial phase, we adopt the information theoretic quantity $I(X;Z|Y)$, where random variable X is input data, Z is the representation produced by model given X, and $Y\in\{1,2,...,C\}$ is the label of X. $I(X;Z|Y)$ is the conditional mutual information, which characterizes, how much information is left in Z about X when conditioned on knowing Y. Therefore, larger $I(X;Z|Y)$ implies that more information about X are preserved in Z besides the purpose of discriminating classes in $\{1,2,...,C\}$.

In addition, following the exposition in the main text, we denote the covariance matrix of $Z|Y=c$ as $K^{(c)}$. The number of classes at the initial phase is denoted by C. We view Z and X as discrete random variable[1]. In addition, we assume that $P_{Z|Y}(z|c)$ **is a quantized Gaussian distribution**. That is to say, although $Z|y=c$ is a discrete random variable, its probability mass function is infinitely close to the density function of a **Gaussian distribution** $p_{Z|Y}(z|c)$. Therefore, we shall have: $P_{Z|Y}(z|c)=p_{Z|Y}(z|c)\delta$.

To start with, we study $I(X;Z|Y=c)$, which is the mutual information between X and Z conditioned on Y being a specific class c:

$$\begin{aligned}
I(X;Z|Y=c) &= H(Z|Y=c)-H(Z|X,Y=c)\\
&= H(Z|Y=c)+0 \qquad (Z \text{ is deterministic conditioned on } X.)\\
&= \sum_{z\in\mathcal{Z}} P_{Z|Y}(z|c)\delta\log\frac{1}{P_{Z|Y}(z|c)\delta}\\
&\approx \int_{\mathcal{Z}} -p_{Z|Y}(z|c)\log p_{Z|Y}(z|c)+\log\frac{1}{\delta}\\
&= a\log\det(K^{(c)})+b \qquad (\text{closed-form solution of differential entropy on Gaussian variables.})\\
&= a\sum_{i=1}^{d}\log\lambda_i^{(c)}+b,
\end{aligned} \tag{4}$$

where a, b are positive constant. Furthermore, we assume that probability of one sample being any class are the same, which

[1]This is reasonable because as we know, each pixel of an image is represented by $3\times 8=24$ bits, which makes an image a discrete random variable.

further leads to:

$$\begin{aligned} I(X;Z|Y) &= \sum_{c=1}^{C} p(Y=c) I(X;Z|Y=c) \\ &= a\frac{1}{C}\sum_{c=1}^{C}\sum_{i=1}^{d}\log\lambda_i^{(c)} + b. \end{aligned} \tag{5}$$

To this end, we have obtained an estimation on $I(X;Z|Y)$ given Eqn. (5).

Based on what we have derived in Eqn. (5), we can visualize $\frac{1}{C}\sum_{c=1}^{C}\sum_{i=1}^{d}\log\lambda_i^{(c)}$ as an proxy of $I(X;Z|Y)$. In this way, we can get to know how $I(X;Z|Y)$ varies for model trained with different CwD coefficient (i.e., η in eq. (9) of main text). Results are shown in Fig. 1.

As can be seen, with larger η, $I(X;Z|Y)$ consistently increase. This means that as we increase η, representation Z will contain more information about X besides information useful for classification at initial phase. Although these additional information are not useful for classification at initial phase, they could be useful for discriminating between classes of initial phase and future phases. In this way, CwD might be beneficial for incrementally learning new classes.

We leave more systematic and rigorous studies on why CwD can help CIL as future works.

C. Analysis on Why CwD Coefficient Being Too Large is Bad (Main text Sec. 3.3)

In the third ablation study of Sec. 3.3 of main text, we ablate how the CwD Coefficient (i.e., η in eq. (9) of main text) affect the improvements brought by CwD. From the results, we find that as we increase η, performance gain brought by CwD will eventually decrease when η is too large.

To understand this phenomenon, we define the volume occupied by representations of class c as $V^{(c)}$. By assuming representations of class c are Gaussian distributed, we have:

$$V^{(c)} \propto \prod_{i=1}^{d}\lambda_i^{(c)}, \tag{6}$$

where $(\lambda_1^{(c)}, ..., \lambda_d^{(c)})$ are eigenvalues of the covariance matrix of representations of class c. This is based on the geometric interpretation of eigenvalues of covariance matrix. Further, combining Eqn. (6) with Eqn. (4), we have:

$$\log V^{(c)} \propto \sum_{i=1}^{d}\log\lambda_i^{(c)} \propto I(X;Z|Y=c). \tag{7}$$

Based on Eqn. (7) and the observation in Sec. B that $I(X;Z|Y=c)$ consistently increase with increasing η, we know that representations of class c will occupy more volume in representation space when η increase. Therefore, when η is too large, representations of initial phase classes will occupy too much space, causing large overlaps with classes in future phases.

D. Extended Visualization on Figure 1 of Main Text

More Details on Setups in Figure 1 of Main Text. For the two-phase CIL setting, each phase contains 2 classes from CIFAR100. We train an AlexNet model for visualization. The representation dimension of AlexNet is set to 3 for convenience of visualization. Representations are normalized to unit sphere as done in LUCIR [2].

Extended Visualizations. Here, we extended visualization of Figure 1 (a) and Figure 1 (d) of main text. Although for these two figures, **the model is only trained on the 2 classes of initial phase**, we visualize data representations of all four classes. Results are shown in Fig. 2.

Surprisingly, from the figure, we can see that when naïvely traiend on the 2 classes of initial phase, data representations of the other 2 classes (i.e., class 3 and class 4) also only lies in the long and narrow region between representations of class 1 and class 2. However, after applying our CwD, data representations of class 3 and class 4 are also more uniformly scattered in representation space. These phenomenon further suggest that applying our CwD can enforce representation to preserve more information about input data, and thus benefit CIL.

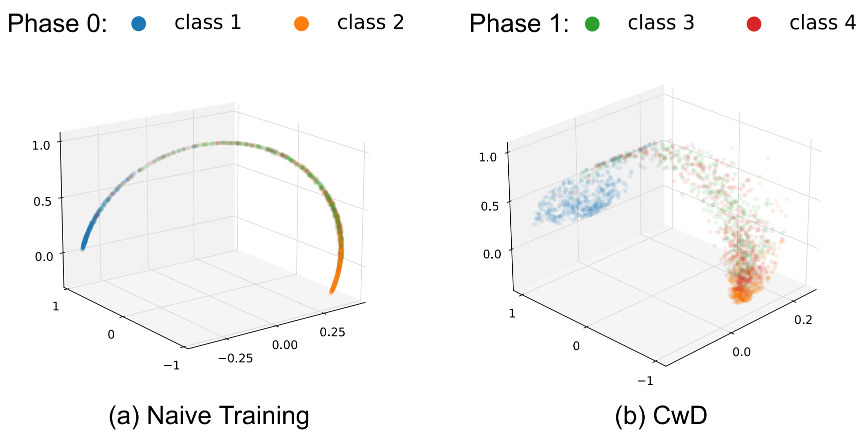

Figure 2. Visualization of representations (normalized to the unit sphere) in a two-phase CIL setting (learning 2 classes for each phase). **(a)** Naïve training at the initial phase (a.k.a., phase 0) but visualize on data of both phase 0 and phase 1 (i.e., including all 4 classes). The model has not been trained on class 3 and class 4, but data representations of these class still lie in a long and narrow region between representations of class 1 and class 2. **(b)** Training with CwD at the initial phase (a.k.a., phase 0) but visualize on data of both phase 0 and phase 1. With our CwD, not only representations of class 1 and class 2 scatter more uniformly, representations of class 3 and class 4 also scatter more uniformly in the space.

E. More Visualizations on Other Classes (Main text Sec. 2.2)

In the Sec. 2.2 of main text, based on ImageNet100, we generate four subsets containing 10/25/50/100 classes, where the subset with more classes contains the subset with fewer classes (the 10 classes of the first subset are shared by all 4 subsets). Based on these four subsets, we train four ResNet18 models. Next, given the four ResNet18 models trained on different number of classes, we visualize $\alpha_k^{(c)}$ for representations of one of the 10 shared classes. Here, we provide the visualization for the other 9 classes. The results are shown in Fig. 3. As can be observed, for the other 9 shared classes, curve of $\alpha_k^{(c)}$ show the similar trend as class 1 shown in the main text, i.e., for every fixed k, $\alpha_k^{(c)}$ consistently decrease as the model being trained with more classes. These results further validated our observations mentioned in main text Sec. 2.2: as the model being jointly trained with more number of classes, representations of each class scatter more uniformly.

F. Comparison with Other Decorrelation Methods.

As we mentioned in the main text, there are some works in other fields that utilized feature decorrelation for other purposes (e.g. boosting generalization). Here, we provide comparison between CwD and some of other decorrelation methods [1, 3] under CIL setting.

Specifically, based on LUCIR, we first use a ResNet18 to learn 50 classes of ImageNet100 at the initial phase, and then learn 10 new classes per phase. We apply DC [1], SDC [3], and CwD at the initial phase, respectively (see Tab. 1). Our results show that although adding DC and SDC at initial phase can also boost CIL performance due to their decorrelation effects, the improvement brought by CwD is larger. The advantage of CwD is possibly because treating each class separately can better mimic representations of oracle model.

Methods	LUCIR	+DC	+SDC	**+CwD (Ours)**
Acc.	70.60±0.43	71.39±0.21	71.52±0.40	**71.94**±0.11

Table 1. **Comparing CwD with other decorrelation methods.** Average Incremental Accuracy (%) is reported. All results (mean±std) are averaged over 3 runs.

G. Why not apply CwD in "later phases"?

Based on our observations, adding CwD to the later phases is unnecessary. This is because once we apply CwD at the initial phase, representations of later phase classes will automatically be decorrelated. To see this, we apply eigenvalue

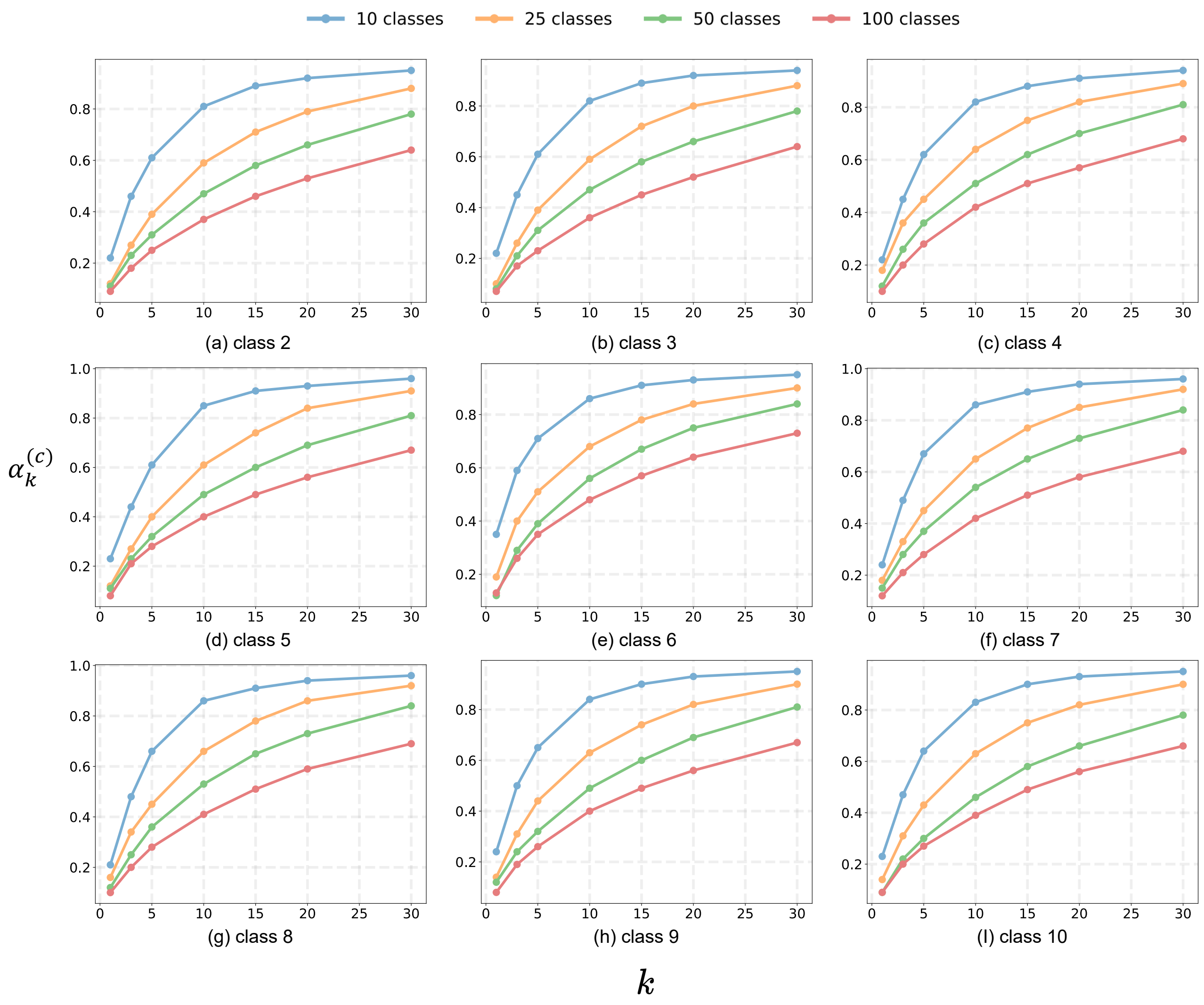

Figure 3. **Visualization on how $\alpha_k^{(c)}$ changes with increasing k for models trained with different number of classes.** $\alpha_k^{(c)}$ curve generated by the model trained on 10/25/50/100 classes are denoted by blue/orange/green/red, respectively. From left to right, top to bottom are figures based on representations of class 2 to 10, respectively. The figure generated by representations of class 1 is shown in the main text.

analysis as in Sec. 2.2 of our paper. We analyze representations of **a class learned at a later phase** with the following three models: 1) naïve LUCIR model incrementally learned all classes; 2) LUCIR with CwD at initial phase; 3) oracle model. Results are shown in Fig. 4. Our results show that even though we **only apply CwD at initial phase**, representations of the **class learned at later phases** will also mimic oracle model.

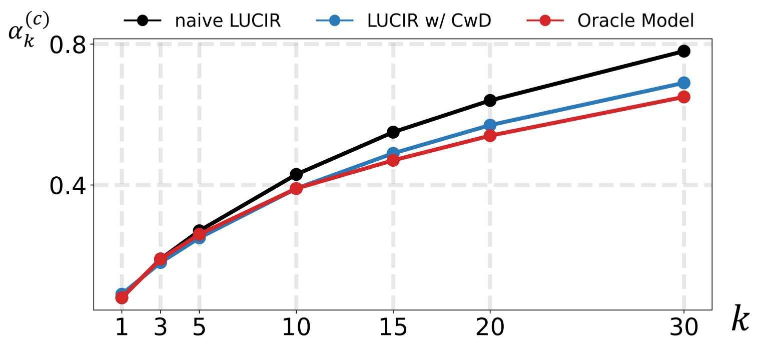

Figure 4. Visualization of how $\alpha_k^{(c)}$ defined in Eqn. (3) (of the main paper) varies with increasing k for different models.

H. What If Keep Increasing β in Fig. 2 in Main Paper.

According to our experiments, $\beta > 15$ (e.g. $\beta = 30$) yields similar performance as $\beta = 15$. Results are shown in Fig. 5.

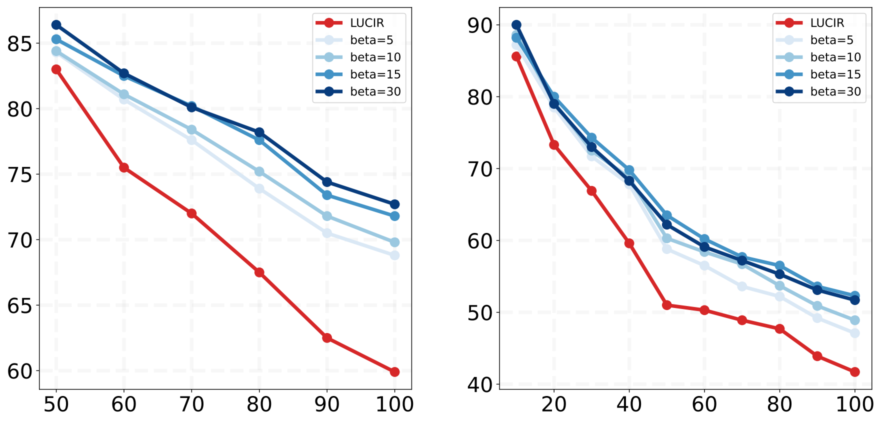

Figure 5. Exploratory experiments the same as Fig. 2 in main paper. $\beta = 30$ is added comparing to Fig. 2 of main paper. As one can observe, $\beta = 30$ yields approximately the same results as $\beta = 15$.

References

[1] Michael Cogswell, Faruk Ahmed, Ross Girshick, Larry Zitnick, and Dhruv Batra. Reducing overfitting in deep networks by decorrelating representations. *arXiv preprint arXiv:1511.06068*, 2015.

[2] Saihui Hou, Xinyu Pan, Chen Change Loy, Zilei Wang, and Dahua Lin. Learning a unified classifier incrementally via rebalancing. In *Proceedings of the IEEE/CVF conference on Computer Vision and Pattern Recognition*, pages 831–839, 2019.

[3] Wei Xiong, Bo Du, Lefei Zhang, Ruimin Hu, and Dacheng Tao. Regularizing deep convolutional neural networks with a structured decorrelation constraint. In *2016 IEEE 16th International Conference on Data Mining (ICDM)*, pages 519–528. IEEE, 2016.