Dataset Knowledge Transfer for Class-Incremental Learning without Memory

Habib Slim[1]* Eden Belouadah[1,2]* Adrian Popescu[1] Darian Onchis[3]
[1] Université Paris-Saclay, CEA, List, F-91120, Palaiseau, France
[2] IMT Atlantique, Lab-STICC, team RAMBO, UMR CNRS 6285, F-29328, Brest, France
[3] West University of Timisoara, Timisoara, Romania
habib.slim@grenoble-inp.org, {eden.belouadah, adrian.popescu}@cea.fr, darian.onchis@e-uvt.ro

Abstract

Incremental learning enables artificial agents to learn from sequential data. While important progress was made by exploiting deep neural networks, incremental learning remains very challenging. This is particularly the case when no memory of past data is allowed and catastrophic forgetting has a strong negative effect. We tackle class-incremental learning without memory by adapting prediction bias correction, a method which makes predictions of past and new classes more comparable. It was proposed when a memory is allowed and cannot be directly used without memory, since samples of past classes are required. We introduce a two-step learning process which allows the transfer of bias correction parameters between reference and target datasets. Bias correction is first optimized offline on reference datasets which have an associated validation memory. The obtained correction parameters are then transferred to target datasets, for which no memory is available. The second contribution is to introduce a finer modeling of bias correction by learning its parameters per incremental state instead of the usual past vs. new class modeling. The proposed dataset knowledge transfer is applicable to any incremental method which works without memory. We test its effectiveness by applying it to four existing methods. Evaluation with four target datasets and different configurations shows consistent improvement, with practically no computational and memory overhead.

1. Introduction

Incremental learning (IL) enables the adaptation of artificial agents to dynamic environments in which data is presented in streams. This type of learning is needed when access to past data is limited or impossible, but is affected by catastrophic forgetting [21]. This phenomenon consists in a drastic performance drop for previously learned information when ingesting new data. Works such as [4, 8, 12, 24, 29, 30, 31] alleviate the effect of forgetting by replaying past data samples when updating deep incremental models in class IL. A term which adapts knowledge distillation [11] to IL is usually exploited to reinforce the representation of past classes [18]. When such a memory is allowed, class IL actually becomes an instance of imbalanced learning [10]. New classes are favored since they are represented by a larger number of images. As a result, classification bias correction methods were successfully introduced in [4, 29, 30].

While important progress was made when a fixed memory is allowed, this is less the case for class IL without memory. This last setting is more challenging and generic since no storage of past samples is allowed. In absence of memory, existing methods become variants of *Learning without Forgetting* (LwF) [18] with different formulations of the distillation term. Importantly, bias correction methods become inapplicable without access to past classes samples.

Our main contribution is to enable the use of the bias correction methods, such as the *BiC* layer from [29], in class IL without memory. We focus on this approach because it is both simple and effective in IL with memory [2, 20]. Authors of *BiC* [29] use a validation set which stores samples of past classes to optimize parameters. Instead, we learn correction parameters offline on a set of reference datasets and then transfer them to target datasets. The method is thus abbreviated *TransIL*. The intuition is that, while datasets are different, optimal bias correction parameters are stable enough to be transferable between them. We illustrate the approach in Figure 1, with the upper showing the IL process with a reference dataset. A memory for the validation samples needed to optimize the bias correction layer is allowed since the training is done offline. The lower part of the figure presents the incremental training of a target dataset. The main difference with the standard memoryless IL training comes from the use of a bias correction layer optimized on the reference dataset. Its introduction leads to an improved comparability of prediction scores for past and new classes. Note that the proposed method is applicable to any class IL method, since it only requires the availability of raw predic-

[*] Equal contribution

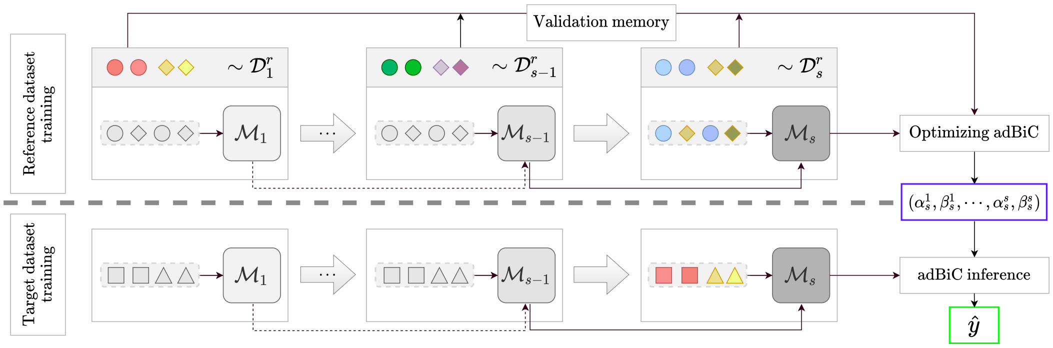


Figure 1: Illustration of *TransIL*, our proposed method, depicting states from 1 to s for a reference and a target dataset. The model $\mathcal{M}$ is updated in each state with data from new classes. States from 1 to $s-1$ are faded to convey the fact that knowledge learned in them is affected by catastrophic forgetting. The class IL process is first launched offline on the reference dataset where *adBiC*, our proposed bias correction layer, is trained using a validation memory which stores samples for past and new classes. Class IL is then applied to the target dataset, but without class samples shared across states since a memory is not allowed in this scenario. The set of optimal parameters of *adBiC* obtained for the reference dataset is transferred to the target dataset. This is the only information shared between the two processes and it has a negligible memory footprint. The transfer of parameters enables the use of bias correction for the target dataset. The final predictions obtained in state s are improved compared to the direct use of $\mathcal{M}_s$ predictions, since the bias in favor of new classes is reduced.

tions provided by deep models $\mathcal{M}_s$.

The second contribution is to refine the definition of the bias correction layer introduced in [29]. The original formulation considers all past classes equally in the correction process. With [20], we hypothesize that the degree of forgetting associated to past classes depends on the initial state in which they were learned. Consequently, we propose *Adaptive BiC* (*adBiC*), an optimization procedure which learns a pair of parameters per IL state instead of a single pair of parameters as proposed in [29].

We provide a comprehensive evaluation of *TransIL* by applying it to four backbone class IL methods. Four target datasets with variable domain shift with respect to reference datasets and different numbers of IL states are used. An improvement of accuracy is obtained for almost all tested configurations. The additional memory needs are negligible since only a compact set of correction parameters is stored. Code and data needed for reproducibility are provided[1].

2. Related work

Incremental learning is a longstanding machine learning task [9, 19, 26] which witnessed a strong growth in interest after the introduction of deep neural networks. It is named differently as continual, incremental or lifelong learning depending on the research communities which tackle it and the setting of the problem. However, the objective is common: enable artificial agents to learn from data which is fed sequentially to them. Detailed reviews of existing approaches are proposed, among others, in [2, 17, 20, 22]. Here, we analyze works most related to our proposal, which tackle class IL and keeps memory and computational requirements constant, or nearly so, during the IL process. We focus particularly on methods which address bias in favor of new classes [20] and were designed for class IL with memory.

The wide majority of class IL methods make use of an information preserving penalty [7]. This penalty is generally implemented as a loss function which reduces the divergence between the current model and the one learned in the preceding IL state. Learning without forgetting (*LwF*) [18] is an early work which tackles catastrophic forgetting in deep neural nets. It exploits knowledge distillation [11] to preserve information related to past classes during incremental model updates. Less-forgetting learning [14] is a closely related method. Past knowledge is preserved by freezing the softmax layer of the source model and updating the model using a loss which preserves the representation of past data. The two methods aim to propose a good compromise between plasticity, needed for new data representation, and stability, useful for past information preservation. However, they require the storage of the preceding model in order to perform distillation toward the model which is currently learned. This requirement can be problematic if the memory of artificial agents is constrained.

LwF was initially used for task-based continual learning and was then widely adopted as backbone for class IL. *iCaRL* [24] exploits *LwF* and a fixed-size memory of past samples to alleviate catastrophic forgetting. In addition, a nearest-mean-of-exemplars classifier is introduced in order to reduce the bias in favor of new classes. *E2EIL* [4] corrects bias by adding a second fine-tuning step with the same number of samples for each past and new class. The learning of a unified classifier for incremental learning re-

[1] `https://github.com/HabibSlim/DKT-for-CIL`

balancing (*LUCIR*) is proposed in [12]. The authors introduce a cosine normalization layer in order to make the magnitudes of past and new class predictions more comparable. The maintenance of both discrimination and fairness is addressed in [30]. The ratio between the mean norm of past and new class weights is applied to the weights of new classes, to make their associated predictions more balanced. Bias Correction (*BiC*) [29] exploits a supplementary linear layer to rebalance predictions of a deep incremental model. A validation set is used to optimize the parameters of this linear layer, which modifies the predictions of the deep model learned in a given incremental state. We tackle two important limitations of existing bias correction methods. First, they are inapplicable without memory because they require the presence of past class samples. We propose to transfer bias correction layer parameters between datasets to address this problem. Second, the degree of forgetting associated to past classes is considered equivalent, irrespective of the initial state in which they were learned. This is problematic insofar as a recency bias, which favors classes more recently, appears in class IL [20]. We refine the linear layer from [29] to improve the handling of recency bias.

The improvement of the component which handles model stability also received strong attention in class IL. Learning without memorizing [7] is inspired by *LwF* and adds an attention mechanism to the distillation loss. This new term improves the preservation of information related to base classes. A distillation component which exploits information from all past states and from intermediate layers of CNN models was introduced in [31]. *LUCIR* [12] distills knowledge in the embedding space rather than the prediction space to reduce forgetting and adds an inter-class separation component to better distinguish between past and new class embeddings. *PODNet* [8] employs a spatial-based distillation loss and a representation which includes multiple proxy vectors for classes to optimize distillation. In [27], a feature map transformation strategy with additional network parameters is proposed to improve class separability. Model parameters are shared between global and task-specific parameters and only the latter are updated at each IL state to improve training times. Feature transformation using a dedicated MLP is introduced in [13]. This approach only stores features but adds significant memory to store the additional MLP. Recently, the authors of [16] argued for the importance of uncertainty and of attention mechanisms in the modeling of past information in class IL. These different works provide a performance gain compared to the original adaptation of distillation for continual learning [18] in class IL with memory.

The utility of distillation in a class IL scenario was recently questioned. It is shown [20, 23] that competitive results are obtained if a fixed-size memory is allowed for large-scale datasets. The distillation component is removed in [20] and IL models are updated using fine-tuning. A simpler approach is tested in [23], where the authors learn models independently for each incremental state after balancing class samples. The usefulness of distillation was also challenged in absence of a memory [1] where standardization of initial weights (*SIW*), learned when a class was first encountered, was proposed in [1]. The freezing of initial weights was tested in [20] and also provides significant improvements. It is thus interesting to also apply the proposed approach to methods which do not exploit distillation.

Our method is globally inspired by existing works which transfer knowledge between datasets. We mentioned knowledge distillation [11] which is widely used in IL. Dataset distillation [28] encodes large datasets into a small set of synthetic data points to make the training process more efficient. Hindsight anchor learning [5] learns an anchor per class to characterize points which would maximize forgetting in later IL states. While the global objective is similar, our focus is different since only a very small number of parameters are transferred from reference to target datasets to limit catastrophic forgetting on the latter.

3. Dataset knowledge transfer for class IL

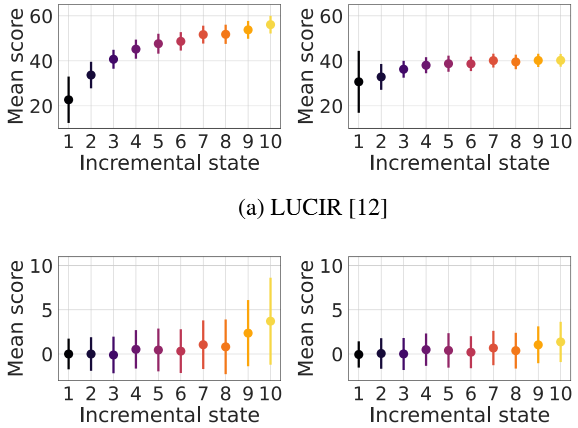
In this section, we describe the proposed approach which transfers knowledge between datasets in class IL without memory. We first propose a formalization of the problem and then introduce an adaptation of a prediction bias correction layer used in class IL with memory. Finally, we introduce the knowledge transfer method which enables the use of the bias correction layer in class IL without memory.

3.1. Class-incremental learning formalization

We adapt the class IL definition from [4, 12, 24] to a setting without memory which includes a sequence of S states. The first one is called initial state and the $S-1$ remaining states are incremental. A set of P_s new classes is learned in the s^{th} state. IL states are disjoint and $P_i \cap P_j = \varnothing \; \forall i, j \in \llbracket 1, S \rrbracket, i \neq j$. A model $\mathcal{M}_1$ is initially trained on a dataset $\mathcal{D}_1 = \{(X_1^j, Y_1^j) \colon j \in P_1\}$, where X_1^j and Y_1^j are the sets of training images and their labels. We note N_s the set of all classes seen until the s^{th} state included. Thus, $N_1 = P_1$ initially, and $N_s = N_{s-1} \cup P_s = P_1 \cup P_2 \cup ... \cup P_{s-1} \cup P_s$ for subsequent states. $\mathcal{M}_s$ is updated with an IL algorithm $\mathcal{A}$ using $\mathcal{D}_s = \{(X_s^j, Y_s^j) \colon j \in P_s\}$. $\mathcal{D}_s$ includes only new classes samples, but $\mathcal{M}_s$ is evaluated on all classes seen so far ($j \in N_s$). This makes the evaluation prone to catastrophic forgetting due to the lack of past exemplars [2, 20].

3.2. Adaptive bias correction layer

The unavailability of past class exemplars when updating the incremental models leads to a classification bias toward new classes [29, 30]. We illustrate this in Figure 2 (*left*) by plotting mean prediction scores per state for the CIFAR-100



(a) LUCIR [12]

(b) LwF [18]

Figure 2: Mean prediction scores and standard deviations for CIFAR-100 classes grouped by state at the end of an IL process with $S = 10$ states, for *LwF* and *LUCIR*, before (left) and after (right) calibration with $TransIL$.

dataset with $S = 10$ splits using *LUCIR* and *LwF*, the two distillation-based approaches tested here. Figure 2 confirms that recently learned classes are favored, despite the use of knowledge distillation to counter the effects of catastrophic forgetting. New classes, learned in the last state, are particularly favored. The predictions profiles for *LUCIR* and *LwF* are different. *LUCIR* mean predictions per state increase from earlier to latest states, while the tendency is less clear for *LwF*. *LwF* predictions also have a stronger deviation in each state. These observations make *LUCIR* a better candidate for bias correction compared to *LwF*.

Among the methods proposed to correct bias, the linear layer introduced in [29] is interesting for its simplicity and effectiveness [2]. This layer is defined in the s^{th} state as:

$$BiC(\boldsymbol{o_s^k}) = \begin{cases} \boldsymbol{o_s^k} & \text{if } \ k \in \llbracket 1,\ s-1 \rrbracket \\ \alpha_s \boldsymbol{o_s^k} + \beta_s \cdot \mathbf{1} & \text{if } \ k = s \end{cases} \qquad (1)$$

where $\boldsymbol{o_s^k}$ are the raw scores of classes first seen in the k^{th} state, obtained with $\mathcal{M}_s$; (α_s, β_s) are the bias correction parameters in the s^{th} state, and $\mathbf{1}$ is a vector of ones.

Equation 1 rectifies the raw predictions of new classes learned in the s^{th} state to make them more comparable to those of past classes. The deep model is first updated using $\mathcal{D}_s$ containing new classes for this state. The model is then frozen and calibration parameters (α_s and β_s) are optimized using a validation set made of samples of new and past classes. We remind that Equation 1 is not applicable in class IL without memory, the scenario explored here, because samples of past classes are not allowed. Figure 2 (*left*) shows that mean scores of classes learned in different incremental states are variable, which confirms that the amount of forgetting is uneven across past states. It is important to tune bias correction for classes which were learned in different IL states. We thus define an adaptive version of *BiC* which rectifies predictions in the s^{th} state with:

$$adBiC(\boldsymbol{o_s^k}) = \alpha_s^k \boldsymbol{o_s^k} + \beta_s^k \cdot \mathbf{1}\ ; \quad k \in \llbracket 1, s \rrbracket \qquad (2)$$

where α_s^k, β_s^k are the parameters applied in the s^{th} state to classes first learned in the k^{th} state.

Differently from Equation 1, Equation 2 adjusts prediction scores depending on the state in which classes were first encountered in the IL process. Note that each α_s^k, β_s^k pair is shared between all classes first learned in the same state. These parameters are optimized on a validation set using the cross-entropy loss, defined for one data point $(\mathbf{x}, y)$ as:

$$\mathcal{L}(\boldsymbol{q_s}, y) = -\sum_{k=1}^{s} \sum_{i=1}^{|P_k|} \delta_{y=\widehat{y}} \ \log \left(q_{s,i}^k \right) \qquad (3)$$

where y is the ground-truth label, $\widehat{y}$ is the predicted label, δ is the Kronecker delta, and $\boldsymbol{q_s}$ is the softmax output for the sample corrected via Equation 2, defined as:

$$\boldsymbol{q_s} = \sigma \left(\left[\alpha_s^1 \boldsymbol{o_s^1} + \beta_s^1 \cdot \mathbf{1}\ ;\ \dots\ ; \alpha_s^s \boldsymbol{o_s^s} + \beta_s^s \cdot \mathbf{1} \right] \right) \qquad (4)$$

where σ is the softmax function.

All α_s^k, β_s^k pairs are optimized using validation samples from classes in N_s. We compare *adBiC* over *BiC* for our class IL setting in the evaluation section and show that the adaptation proposed here has a positive effect.

3.3. Transferring knowledge between datasets

The optimization of α and β parameters is impossible in class IL without memory, since exemplars of past classes are unavailable. To circumvent this problem, we hypothesize that optimal values of these parameters can be transferred between reference and target datasets, noted $\mathcal{D}^r$ and $\mathcal{D}^t$ respectively. The intuition is that these values are sufficiently stable despite dataset content variability. We create a set of reference datasets and perform a modified class IL training for them using the procedure described in Algorithm 1. The modification consists in exploiting a validation set which includes exemplars of classes from all incremental states. Validation set storage is necessary in order to optimize the parameters from Equation 2 and is possible since reference dataset training is done offline. Note that backbone incremental models for $\mathcal{D}^r$ are trained without memory in order to simulate the IL setting of target datasets $\mathcal{D}^t$. We then store bias correction parameters optimized for reference datasets in order to perform transfer toward target datasets without using a memory. For each incremental state, we compute the average of α and β values over all reference datasets. The obtained averages are used for score rectification on target datasets. This transfer uses the procedure described in Algorithm 2. The memory needed to store transferred parameters is negligible since we need $2\times(2+3+...+S) = (S+2)\times(S-1)$ floats for each dataset and S value. For $S = \{5, 10, 20\}$ states, we thus only store 28, 108 and 418 floating-point values respectively.

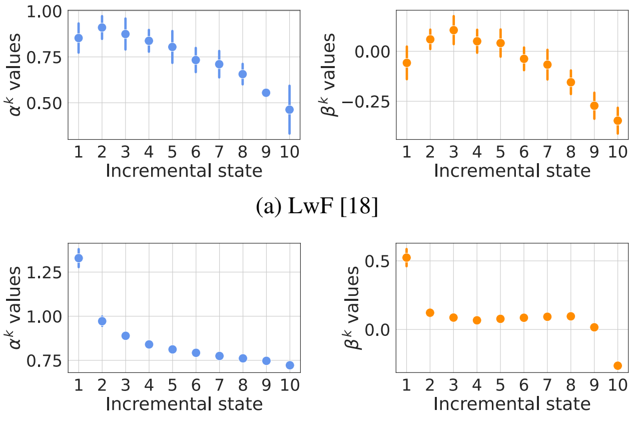
Algorithm 1: Optimization of calibration parameters

```
inputs : $\mathcal{A}, \mathcal{D}_s^r$ for $s \in [\![1, S]\!]$        ▷ reference dataset
randomly initialize $\mathcal{M}_1$ ;
$\mathcal{M}_1^* \leftarrow$ train($\mathcal{A}; \mathcal{M}_1, \mathcal{D}_1^r$) ;
for $s = 2 \ldots S$ do
    $\mathcal{M}_s^* \leftarrow$ update($\mathcal{A}; \mathcal{M}_{s-1}^*, \mathcal{D}_s^r$) ;
    $\alpha_s^k \leftarrow 1,\ \beta_s^k \leftarrow 0$ for each $k \in [\![1, s]\!]$ ;
    foreach $(\mathbf{x}, y) \in \mathcal{D}_s^r$        ▷ validation set
     do
        $\mathbf{o_s} \leftarrow \mathcal{M}_s^*(\mathbf{x})$ ;
        for $k = 1 \ldots s$ do
            $\mathbf{o_s^k} \leftarrow adBiC(\boldsymbol{o_s^k}) = \alpha_s^k \boldsymbol{o_s^k} + \beta_s^k \cdot \mathbf{1}$ ;
        end
        $\mathbf{q_s} \leftarrow \sigma(\mathbf{o_s})$ ;
        loss $\leftarrow \mathcal{L}(\boldsymbol{q_s}, y)$ ;
        $(\alpha_s^1, \beta_s^1, \ldots, \alpha_s^s, \beta_s^s) \leftarrow$ optimize(loss) ;
    end
end
```

Algorithm 2: *adBiC* inference

```
inputs : $\mathcal{A}, (\alpha_s^k, \beta_s^k)$ averaged on reference datasets
             for each $s \in [\![1, S]\!]$, $k \in [\![1, s]\!]$
inputs : $\mathcal{D}_s^t$ for $s \in [\![1, S]\!]$        ▷ target dataset
randomly initialize $\mathcal{M}_1$ ;
$\mathcal{M}_1^* \leftarrow$ train($\mathcal{A}; \mathcal{M}_1, \mathcal{D}_1^t$);
for $s = 2 \ldots S$ do
    $\mathcal{M}_s^* \leftarrow$ update($\mathcal{A}; \mathcal{M}_{s-1}^*, \mathcal{D}_s$) ;
    foreach $(\mathbf{x}, y) \in \mathcal{D}_s^t$        ▷ test set
     do
        $\mathbf{o_s} \leftarrow \mathcal{M}_s^*(\mathbf{x})$ ;
        for $k = 1 \ldots s$ do
            $\mathbf{o_s^k} \leftarrow adBiC(\boldsymbol{o_s^k}) = \alpha_s^k \boldsymbol{o_s^k} + \beta_s^k \cdot \mathbf{1}$ ;
        end
        $\mathbf{q_s} \leftarrow \sigma(\mathbf{o_s})$ ;
        $\hat{y} \leftarrow \underset{y \in [\![1, N_s]\!]}{argmax}(\mathbf{q_s})$ ;        ▷ inference
    end
end
```

In Figure 3, we illustrate optimal parameters obtained across $R = 10$ reference datasets which are further described in Section 4. We plot α^k and β^k values learned after $S = 10$ IL states, using *LwF* [18] and *LUCIR* [12] methods. Mean and standard deviations are presented for past and current incremental states in the final state of the IL process. The parameter ranges from Figure 3 confirm that, while optimal values do vary across datasets, this variation is rather low and calibration profiles remain similar. Consequently, parameters are transferable. When $R > 1$, a transfer function is needed to apply the parameters learned on reference datasets to a target dataset. We transfer parameters using the averaged α_s^k and β_s^k values, obtained for the set of $\mathcal{D}^r$. In Section 4, we evaluate this transfer against an upper-bound oracle which selects the best $\mathcal{D}^r$ in each state.



(a) LwF [18]

(b) LUCIR [12]

Figure 3: Averaged α^k (left) and β^k (right) values computed for $R = 10$ reference datasets using *LwF* and *LUCIR*, at the end of an incremental process with $S = 10$ states.

The proposed approach adds a simple but effective linear layer to calibrate the predictions of backbone class IL methods. Consequently, it is applicable to any IL method which works without memory. We test the genericity of the approach by applying it on top of four existing methods.

4. Evaluation

In this section, we discuss: (1) the reference and target datasets, (2) the backbone methods to which bias correction is applied and (3) the analysis of the obtained results. The evaluation metric is the average top-1 accuracy of the IL process introduced in [24], which combines the accuracy obtained for individual incremental states. Following [4], we discard the accuracy of the first state since it is not incremental. We use a ResNet-18 backbone whose implementation details are provided in the supp. material.

4.1. Datasets

Reference datasets. The preliminary analysis from Figure 3 indicates that bias correction parameters are rather stable for different reference datasets. It is interesting to use several such datasets in order to stabilize averaged bias correction parameters. In our experiments, we use 10 reference datasets, each including 100 randomly chosen leaf classes from ImageNet [6], with a 500/200 train/val split per class. There is no intersection between these datasets, as each class appears only in one of them.

Target datasets. We test *TransIL* with four target datasets. They were selected to include different types of visual content and thus test the robustness of the parameter transfer. The class samples from the target datasets are split into 500/100 train/test subsets respectively. There is no intersection between the classes from the reference datasets and the two target datasets which are sampled from ImageNet. We describe target datasets briefly hereafter and provide details in the supplementary material:

- CIFAR-100 [15] - object recognition dataset. It focuses on commonsense classes and is relevant for basic level classification in the sense of [25].
- IMN-100 - subset of ImageNet [6] which includes 100 randomly selected leaf classes. It is built with the same procedure used for reference datasets and is thus most similar to them. IMN-100 is relevant for fine-grained classification with a diversity of classes.
- BIRDS-100 - uses 100 bird classes from ImageNet [6]. It is built for domain fine-grained classification.
- FOOD-100 - uses 100 food classes from Food-101 [3]. It is a fine-grained and domain-specific dataset and is interesting because it is independent from ImageNet.

4.2. Backbone incremental learning methods

We apply *adBiC* on top of four backbone methods which are usable for class IL without memory:

- *LwF* [24] - version of the original method from [18] which exploits distillation to reduce catastrophic forgetting for past classes.
- *LUCIR* [12] - distillation-based approach which uses a more elaborate way of ensuring a good balance between model stability and plasticity. We use the CNN version because it is adaptable to our setting.
- *FT+* [20] - fine-tuning in which past classes weights are not updated to reduce catastrophic forgetting.
- *SIW* [1] - similar to *FT+*, but with class weights standardization added to improve the comparability of prediction between past and new classes.

We compare *adBiC* to *BiC*, the original linear layer from [29]. We also provide results with an optimal version of *adBiC*, which is obtained via an oracle-based selection of the best performing reference dataset for each IL state. This oracle is important as it indicates the potential supplementary gain obtainable with a parameter selection method more refined than the proposed one. Finally, we provide results with *Joint*, a training from scratch with all data available at all times. This is an upper bound for all IL methods.

4.3. Overall results

Results from Figure 2 (*right*) indicate that the degree of forgetting depends on the initial state in which classes were first learned. Applying calibration parameters learned on reference datasets clearly reduces the imbalance of mean prediction scores and the bias toward recent classes.

Results from Table 1 show that our method improves the performance of baseline methods for all but two of the configurations evaluated. The best overall performance before bias correction is obtained with *LwF*. This result confirms the conclusions of [1, 20] regarding the strong performance of *LwF* in class IL without memory for medium-scale datasets. With *adBiC*, *LUCIR* performs generally better than *LwF* for $S = 5$ and $S = 10$, while *LwF* remains stronger with $S = 20$ states. Results are particularly interesting for *LUCIR*, a method for which *adBiC* brings consistent gains (up to 16 accuracy points) in most configurations. Table 1 shows that *adBiC* also improves the results of *LwF* in all configurations, albeit to a lesser extent compared to *LUCIR*. Interestingly, improvements for *LwF* are larger for $S = 20$ states. This is the most challenging configuration since the model is more prone to forgetting. *FT+* [20] and *SIW* [1] remove the distillation component for the class IL training process and exploit the weights of past classes learned in their initial state. *adBiC* improves results for these two methods in all but one configuration. However, their global performance is significantly lower than that of *LwF* and *LUCIR*, the two methods which make use of distillation. This result confirms the finding from [1] regarding the usefulness of the distillation term exploited by *LwF* and *LUCIR* to stabilize IL training for medium scale datasets.

Results from Table 1 highlight the effectiveness of *adBiC* compared to *BiC*. *adBiC* has better accuracy in all tested configurations, with the most important gain over *BiC* obtained for *LUCIR*. It is also worth noting that *adBiC* improves results for *SIW* and *FT+* in most configurations, while the corresponding results of *BiC* are mixed. The comparison of *adBiC* and *BiC* validates our hypothesis that a finer-grained modeling of forgetting for past states is better compared to a uniform processing of them. It would be interesting to test the usefulness of *adBiC* in the class IL with memory setting originally tested in [29].

We also compare *adBiC*, which uses averaged α and β parameters, with an oracle selection of parameters (+ $\mathbb{O}$). The performance of *adBiC* is close to this upper bound for all tested methods This indicates that averaging parameters is an effective way to aggregate parameters learned from reference datasets. However, it would be interesting to investigate more refined ways to transfer parameters from reference to target datasets to further improve performance.

The comparison of target datasets shows that the gain brought by *adBiC* is the largest for IMN-100, followed by BIRDS-100, CIFAR-100 and FOOD-100. This is intuitive

Method	CIFAR-100			IMN-100			BIRDS-100			FOOD-100		
	$S = 5$	$S = 10$	$S = 20$	$S = 5$	$S = 10$	$S = 20$	$S = 5$	$S = 10$	$S = 20$	$S = 5$	$S = 10$	$S = 20$
LwF [18]	53.0	44.0	29.1	53.8	41.1	29.2	53.7	41.8	30.1	42.9	31.8	22.2
w/ BiC	54.0 + 1.0	45.5 + 1.5	30.8 + 1.7	54.7 + 0.9	42.5 + 1.4	31.1 + 1.9	54.6 + 0.9	43.1 + 1.3	31.8 + 1.7	43.4 + 0.5	32.6 + 0.8	23.8 + 1.6
w/ adBiC	54.3 + 1.3	**46.4** + 2.4	**32.3** + 3.2	55.1 + 1.3	43.4 + 2.3	**32.3** + 3.1	55.0 + 1.3	44.0 + 2.2	**32.8** + 2.7	43.5 + 0.6	33.3 + 1.5	**24.7** + 2.5
w/ adBiC + $\mathbb{O}$	54.9 + 1.9	47.3 + 3.3	32.6 + 3.5	55.9 + 2.1	44.2 + 3.1	33.1 + 3.9	55.8 + 2.1	44.8 + 3.0	33.3 + 3.2	44.0 + 1.1	34.2 + 2.4	25.3 + 3.1
LUCIR [12]	50.1	33.7	19.5	48.3	30.1	17.7	50.8	31.4	17.9	44.2	26.4	15.5
w/ BiC	52.5 + 2.4	37.1 + 3.4	22.4 + 2.9	54.9 + 6.6	36.8 + 6.7	21.8 + 4.1	56.0 + 5.2	37.7 + 6.3	20.6 + 2.7	49.9 + 5.7	31.5 + 5.1	17.2 + 1.7
w/ adBiC	**54.8** + 4.7	42.2 + 8.5	28.4 + 8.9	**59.0** + 10.7	**46.1** + 16.0	27.3 + 9.6	**58.5** + 7.7	**45.4** + 14.0	27.3 + 9.4	**52.0** + 7.8	**37.1** + 10.7	17.7 + 2.2
w/ adBiC + $\mathbb{O}$	55.5 + 5.4	43.6 + 9.9	31.2 + 11.7	59.4 + 11.1	46.6 + 16.5	29.7 + 12.0	59.0 + 8.2	46.0 + 14.6	28.8 + 10.9	52.6 + 8.4	38.2 + 11.8	21.0 + 5.5
SIW [1]	29.9	22.7	14.8	32.6	23.3	15.1	30.6	23.2	14.9	29.4	21.6	14.1
w/ BiC	31.4 + 1.5	22.8 + 0.1	14.7 - 0.1	33.9 + 1.3	22.6 - 0.7	13.9 - 1.2	32.8 + 2.2	22.7 - 0.5	12.8 - 2.1	29.1 - 0.3	20.3 - 1.3	12.1 - 2.0
w/ adBiC	31.7 + 1.8	24.1 + 1.4	15.8 + 1.0	35.1 + 2.5	24.5 + 1.2	15.0 - 0.1	33.0 + 2.4	25.2 + 2.0	15.3 + 0.4	30.9 + 1.5	21.3 - 0.3	14.5 + 0.4
w/ adBiC + $\mathbb{O}$	32.8 + 2.9	25.0 + 2.3	16.5 + 1.7	36.4 + 3.8	25.7 + 2.4	16.1 + 1.0	34.4 + 3.8	26.2 + 3.0	16.3 + 1.4	31.5 + 2.1	22.6 + 1.0	15.1 + 1.0
FT+	28.9	22.6	14.5	31.7	23.2	14.6	29.7	23.3	13.5	28.7	21.1	13.3
w/ BiC	30.7 + 1.8	22.5 - 0.1	14.8 + 0.3	33.0 + 1.3	21.9 - 1.3	13.8 - 0.8	32.3 + 2.6	22.5 - 0.8	12.4 - 1.1	28.6 - 0.1	20.6 - 0.5	11.8 - 1.5
w/ adBiC	31.9 + 3.0	23.6 + 1.0	15.0 + 0.5	34.9 + 3.2	23.7 + 0.5	15.7 + 1.1	34.0 + 4.3	25.0 + 1.7	14.2 + 0.7	30.8 + 2.1	22.2 + 1.1	14.2 + 0.9
w/ adBiC + $\mathbb{O}$	32.5 + 3.6	24.6 + 2.0	15.9 + 1.4	35.7 + 4.0	24.9 + 1.7	16.2 + 1.6	34.5 + 4.8	25.7 + 2.4	15.4 + 1.9	31.3 + 2.6	22.7 + 1.6	14.5 + 1.2
Joint		72.7			75.5			80.9			71.03	

Table 1: Average top-1 incremental accuracy using $S = \{5, 10, 20\}$ states. Results are presented for each method without parameter transfer and with *BiC* and *adBiC* transfer. The gain (green) and loss (red) in accuracy obtained with parameter transfer are provided for each configuration. *Joint* is an upper bound obtained using a standard training with all data available. $\mathbb{O}$ denotes a choice of the reference dataset by oracle, in which the best reference dataset for each state is selected for transfer. Best results for each setting (excluding the oracle) are in bold. A graphical view of this table is in the supplementary material.

as IMN-100 has the closest distribution to that of reference datasets. BIRDS-100 is extracted from ImageNet and, while topically different from reference datasets, was created using similar guidelines. The consistent improvements obtained with CIFAR-100 and FOOD-100, two datasets independent from ImageNet, shows that the proposed transfer method is robust to data distribution changes. The performance gaps between IL results and $Joint$ are still wide, particularly for larger values of S. This indicates that class IL without memory remains an open challenge.

Except for *LwF*, *adBiC* gains are larger for $S = \{5, 10\}$ compared to $S = 20$. This result is consistent with past findings reported for bias correction methods [20, 29]. It is mainly explained by the fact that the size of validation sets needed to optimize *adBiC* parameters is smaller and thus less representative for larger values of S. A larger number of states leads to a higher degree of forgetting. This makes the IL training process more challenging and also has a negative effect on the usefulness of the bias correction layer.

Figure 2 provides a qualitative view of the effect of *adBiC* for *LwF* and *LUCIR* which complements numerical results from Table 1. The correction is effective since the predictions associated to IL states are more balanced (right), compared to the raw predictions (left). The effect of calibration is particularly interesting for *LUCIR*, where mean prediction scores are balanced for states 3 to 10. We note that bias correction should ideally provide fully balanced mean prediction scores to give equal chances to classes learned in different states. Some variation subsists and is notably due to variable forgetting for past states and to the variable difficulty of learning different visual classes.

4.4. Robustness of dataset knowledge transfer

We complement the results presented in Table 1 with two experiments which further evaluate the robustness of *adBiC*. First, we test the effect of a different number of training images per class for reference and target datasets. We remove 50% of training images for target datasets to test the transferability in this setting. The obtained results, presented in Table 2, indicate that performance gains are systematic for *LwF* and *LUCIR*, albeit lower compared to results in Table 1. Results are more mixed for *SIW* and *FT+*, but *adBiC* still has a positive effect in the majority of cases. This experiment shows that the proposed dataset knowledge transfer approach is usable for reference and target datasets which have a different number of training samples per class. However, maintaining a low difference in dataset sizes is preferable in order to keep the transfer effective.

Second, we assess the robustness of the method with respect to R, the number of available reference datasets. We select the FOOD-100 dataset because it has the largest domain shift with respect to reference datasets and is thus the most suitable for this experiment. We vary R from 1 to 9 and perform transfer with ten random samplings for each R value. Results obtained for *LUCIR* are reported in Table 3. Accuracy levels are remarkably stable for different values of R and significant gains are obtained even when using a single reference dataset. These results confirm that parameter transfer is effective even with few reference datasets, which is interesting considering that the computational cost

Method	CIFAR-100			IMN-100			BIRDS-100			FOOD-100		
	$S = 5$	$S = 10$	$S = 20$	$S = 5$	$S = 10$	$S = 20$	$S = 5$	$S = 10$	$S = 20$	$S = 5$	$S = 10$	$S = 20$
LwF [18]	41.3	33.3	23.3	45.6	33.5	23.8	44.6	34.0	23.2	29.5	23.3	17.3
w/ adBiC	42.1 **+ 0.8**	34.8 **+ 1.5**	25.0 **+ 1.7**	46.7 **+ 1.1**	35.3 **+ 1.8**	25.6 **+ 1.8**	45.5 **+ 0.9**	35.4 **+ 1.4**	25.2 **+ 2.0**	29.9 **+ 0.4**	24.3 **+ 1.0**	18.7 **+ 1.4**
LUCIR [12]	43.5	27.8	16.6	42.9	27.6	17.0	45.2	27.8	16.0	37.9	22.7	13.9
w/ adBiC	48.3 **+ 4.8**	38.5 **+ 10.7**	25.3 **+ 8.7**	54.1 **+ 11.2**	42.4 **+ 14.8**	23.2 **+ 6.2**	52.8 **+ 7.6**	40.9 **+ 13.1**	25.6 **+ 9.6**	45.7 **+ 7.8**	32.6 **+ 9.9**	19.8 **+ 5.9**
SIW [1]	31.7	21.6	13.7	32.1	22.7	14.4	29.7	22.8	14.1	28.4	18.7	13.5
w/ adBiC	33.7 **+ 2.0**	22.5 **+ 0.9**	14.0 **+ 0.3**	35.0 **+ 2.9**	22.6 **- 0.1**	12.2 **- 2.2**	32.1 **+ 2.4**	23.7 **+ 0.9**	13.5 **- 0.6**	29.9 **+ 1.5**	16.9 **- 1.8**	13.3 **- 0.2**
FT+	30.4	21.5	12.9	31.2	22.2	12.0	29.2	22.8	12.2	27.4	18.2	11.6
w/ adBiC	32.0 **+ 1.6**	21.4 **- 0.1**	13.4 **+ 0.5**	34.8 **+ 3.6**	21.2 **- 1.0**	13.7 **+ 1.7**	31.9 **+ 2.7**	23.0 **+ 0.2**	13.6 **+ 1.4**	28.8 **+ 1.4**	16.2 **- 2.0**	12.2 **+ 0.6**

Table 2: Average top-1 IL accuracy with 50% of training images for target datasets. Gains are in green, losses are in red.

***S* = 5**	Raw	$R = 1$	$R = 2$	$R = 3$	$R = 4$	$R = 5$	$R = 6$	$R = 7$	$R = 8$	$R = 9$	$R = 10$
	44.19	51.9 ± 0.4	52.0 ± 0.2	52.1 ± 0.2	52.0 ± 0.1	52.1 ± 0.1	52.0 ± 0.1	52.0 ± 0.1	52.0 ± 0.1	52.0 ± 0.1	52.0
***S* = 10**	Raw	$R = 1$	$R = 2$	$R = 3$	$R = 4$	$R = 5$	$R = 6$	$R = 7$	$R = 8$	$R = 9$	$R = 10$
	26.44	36.7 ± 0.7	36.9 ± 0.4	37.2 ± 0.4	37.2 ± 0.3	37.1 ± 0.2	37.0 ± 0.2	37.0 ± 0.1	37.1 ± 0.0	37.1 ± 0.1	37.1
***S* = 20**	Raw	$R = 1$	$R = 2$	$R = 3$	$R = 4$	$R = 5$	$R = 6$	$R = 7$	$R = 8$	$R = 9$	$R = 10$
	15.47	17.6 ± 1.2	17.5 ± 0.7	17.6 ± 0.7	17.8 ± 0.4	17.5 ± 0.3	17.7 ± 0.4	17.8 ± 0.3	17.6 ± 0.2	17.7 ± 0.1	17.7

Table 3: Average top-1 incremental accuracy of *adBiC*-corrected models trained incrementally on FOOD-100 with *LUCIR*, for $S = \{5, 10, 20\}$ states, while varying the number R of reference datasets. For $R \leq 9$, results are averaged across 10 random samplings of the reference datasets (hence the std values). *Raw* is the accuracy of *LUCIR* without bias correction.

of offline training is also reduced. Results with other methods for CIFAR-100 are provided in the supp. material.

5. Conclusion

We introduced a method which enables the use of bias correction methods for class IL without memory. This IL scenario is challenging, because catastrophic forgetting is very strong in the absence of memory. The proposed method *TransIL* transfers bias correction parameters learned offline from reference datasets toward target datasets. Since reference dataset training is done offline, a validation memory which includes exemplars from all incremental states can be exploited to optimize the bias correction layer. The evaluation provides comprehensive empirical support for the transferability of bias correction parameters. Performance is improved for all but two of the configurations tested, with gains up to 16 top-1 accuracy points. Robustness evaluation shows that parameter transfer is efficient when only a small number of reference datasets is used for transfer. It is also usable when the number of training images per class in target datasets is different from that of available reference datasets. These last two findings are important in practice since the same reference datasets can be exploited in different incremental configurations. A second contribution relates to the modeling of the degree of forgetting associated to past states. While recency bias was already acknowledged [20], no difference was made between past classes learned in different IL states [29]. This is in part due to validation memory constraints which appear when the bias correction layer is optimized during the incremental process. Such constraints are reduced here since reference datasets training is done offline and a refined definition of the *BiC* layer with specific parameters for each past state becomes possible. The comparison of the standard and of the proposed definition of the bias correction layer is favorable to the latter. The reported results encourage us to pursue the work presented here. First, the parameter transfer is done using average values of parameters learned on reference datasets. A finer-grained transfer method will be tested to get closer to the oracle results reported in Table 1. The objective is to automatically select the best reference dataset in each IL state of a target dataset. Second, we exploit an adapted version of a bias correction method which was initially designed for class IL with memory. We will explore the design of methods which are specifically created for class IL without memory. Finally, while distillation-based methods outperformed methods which do not use distillation for the datasets tested here, existing results [1, 20] indicate that the role of distillation diminishes with scale. It would be interesting to verify this finding for our method.

Acknowledgments. This work was supported by the European Commission under European Horizon 2020 Programme, grant number 951911 - AI4Media, and partially supported by the LabEx PERSYVAL-Lab (ANR-11-LABX-0025-01) funded by the French program "Investissements d'Avenir". This publication was made possible by the use of the FactoryIA supercomputer, financially supported by the Ile-de-France Regional Council.

References

[1] Eden Belouadah, Adrian Popescu, and Ioannis Kanellos. Initial classifier weights replay for memoryless class incremental learning. In *British Machine Vision Conference (BMVC)*, 2020.

[2] Eden Belouadah, Adrian Popescu, and Ioannis Kanellos. A comprehensive study of class incremental learning algorithms for visual tasks. *Neural Networks*, 135:38–54, 2021.

[3] Lukas Bossard, Matthieu Guillaumin, and Luc Van Gool. Food-101 – mining discriminative components with random forests. In *European Conference on Computer Vision*, 2014.

[4] Francisco M. Castro, Manuel J. Marín-Jiménez, Nicolás Guil, Cordelia Schmid, and Karteek Alahari. End-to-end incremental learning. In *Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part XII*, pages 241–257, 2018.

[5] Arslan Chaudhry, Albert Gordo, Puneet K Dokania, Philip Torr, and David Lopez-Paz. Using hindsight to anchor past knowledge in continual learning. *arXiv preprint arXiv:2002.08165*, 2020.

[6] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Fei-Fei Li. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2009), 20-25 June 2009, Miami, Florida, USA*, pages 248–255, 2009.

[7] Prithviraj Dhar, Rajat Vikram Singh, Kuan-Chuan Peng, Ziyan Wu, and Rama Chellappa. Learning without memorizing. *CoRR*, abs/1811.08051, 2018.

[8] Arthur Douillard, Matthieu Cord, Charles Ollion, Thomas Robert, and Eduardo Valle. Podnet: Pooled outputs distillation for small-tasks incremental learning. In *Computer vision-ECCV 2020-16th European conference, Glasgow, UK, August 23-28, 2020, Proceedings, Part XX*, volume 12365, pages 86–102. Springer, 2020.

[9] Bernd Fritzke. A growing neural gas network learns topologies. *Advances in neural information processing systems*, 7:625–632, 1994.

[10] Haibo He and Edwardo A. Garcia. Learning from imbalanced data. *IEEE Trans. Knowl. Data Eng.*, 21(9):1263–1284, 2009.

[11] Geoffrey E. Hinton, Oriol Vinyals, and Jeffrey Dean. Distilling the knowledge in a neural network. *CoRR*, abs/1503.02531, 2015.

[12] Saihui Hou, Xinyu Pan, Chen Change Loy, Zilei Wang, and Dahua Lin. Learning a unified classifier incrementally via rebalancing. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 831–839, 2019.

[13] Ahmet Iscen, Jeffrey Zhang, Svetlana Lazebnik, and Cordelia Schmid. Memory-efficient incremental learning through feature adaptation. In *European Conference on Computer Vision*, pages 699–715. Springer, 2020.

[14] Heechul Jung, Jeongwoo Ju, Minju Jung, and Junmo Kim. Less-forgetting learning in deep neural networks. *arXiv preprint arXiv:1607.00122*, 2016.

[15] Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.

[16] Vinod K Kurmi, Badri N Patro, Venkatesh K Subramanian, and Vinay P Namboodiri. Do not forget to attend to uncertainty while mitigating catastrophic forgetting. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 736–745, 2021.

[17] Matthias De Lange, Rahaf Aljundi, Marc Masana, Sarah Parisot, Xu Jia, Ales Leonardis, Gregory G. Slabaugh, and Tinne Tuytelaars. Continual learning: A comparative study on how to defy forgetting in classification tasks. *CoRR*, abs/1909.08383, 2019.

[18] Zhizhong Li and Derek Hoiem. Learning without forgetting. In *European Conference on Computer Vision*, ECCV, 2016.

[19] Thomas Martinetz, Stanislav G. Berkovich, and Klaus Schulten. 'neural-gas' network for vector quantization and its application to time-series prediction. *IEEE Trans. Neural Networks*, 4(4):558–569, 1993.

[20] Marc Masana, Xialei Liu, Bartlomiej Twardowski, Mikel Menta, Andrew D. Bagdanov, and Joost van de Weijer. Class-incremental learning: survey and performance evaluation on image classification, 2021.

[21] Michael Mccloskey and Neil J. Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. *The Psychology of Learning and Motivation*, 24:104–169, 1989.

[22] German Ignacio Parisi, Ronald Kemker, Jose L. Part, Christopher Kanan, and Stefan Wermter. Continual lifelong learning with neural networks: A review. *Neural Networks*, 113, 2019.

[23] Ameya Prabhu, Philip HS Torr, and Puneet K Dokania. Gdumb: A simple approach that questions our progress in continual learning. In *European Conference on Computer Vision*, pages 524–540. Springer, 2020.

[24] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H. Lampert. icarl: Incremental classifier and representation learning. In *Conference on Computer Vision and Pattern Recognition*, CVPR, 2017.

[25] Eleanor Rosch. Principles of categorization. *Concepts: core readings*, 189, 1999.

[26] Nadeem Ahmed Syed, Huan Liu, and Kah Kay Sung. Handling concept drifts in incremental learning with support vector machines. In *Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 317–321, 1999.

[27] Vinay Kumar Verma, Kevin J. Liang, Nikhil Mehta, Piyush Rai, and Lawrence Carin. Efficient feature transformations for discriminative and generative continual learning. *CoRR*, abs/2103.13558, 2021.

[28] Tongzhou Wang, Jun-Yan Zhu, Antonio Torralba, and Alexei A Efros. Dataset distillation. *arXiv preprint arXiv:1811.10959*, 2018.

[29] Yue Wu, Yinpeng Chen, Lijuan Wang, Yuancheng Ye, Zicheng Liu, Yandong Guo, and Yun Fu. Large scale incremental learning. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 374–382, 2019.

[30] Bowen Zhao, Xi Xiao, Guojun Gan, Bin Zhang, and Shu-Tao Xia. Maintaining discrimination and fairness in class incre-

mental learning. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*, pages 13205–13214. IEEE, 2020.

[31] Peng Zhou, Long Mai, Jianming Zhang, Ning Xu, Zuxuan Wu, and Larry S. Davis. M2KD: multi-model and multi-level knowledge distillation for incremental learning. *CoRR*, abs/1904.01769, 2019.