This CVPR 2021 workshop paper is the Open Access version, provided by the Computer Vision Foundation.
Except for this watermark, it is identical to the accepted version;
the final published version of the proceedings is available on IEEE Xplore.

Essentials for Class Incremental Learning

Sudhanshu Mittal Silvio Galesso Thomas Brox
University of Freiburg, Germany
mittal,galessos,brox@cs.uni-freiburg.de

Abstract

Contemporary neural networks are limited in their ability to learn from evolving streams of training data. When trained sequentially on new or evolving tasks, their accuracy drops sharply, making them unsuitable for many real-world applications. In this work, we shed light on the causes of this well known yet unsolved phenomenon – often referred to as catastrophic forgetting – in a class-incremental setup. We show that a combination of simple components and a loss that balances intra-task and inter-task learning can already resolve forgetting to the same extent as more complex measures proposed in literature. Moreover, we identify poor quality of the learned representation as another reason for catastrophic forgetting in class-IL. We show that performance is correlated with secondary class information (dark knowledge) learned by the model and it can be improved by an appropriate regularizer. With these lessons learned, class-incremental learning results on CIFAR-100 and ImageNet improve over the state-of-the-art by a large margin, while keeping the approach simple.

1. Introduction

The ability to learn from continuously evolving data is important for many real-world applications. Latest machine learning models, especially artificial neural networks, have shown great ability to learn the task at hand, but when confronted with a new task, they tend to override the previous concepts. Deep networks suffer heavily from this catastrophic forgetting [19] when trained with a sequence of tasks, impeding continual or lifelong learning.

In this work, we focus on class-incremental learning (class-IL) [23]. It is one of the three scenarios of continual learning as described in [27], where the objective is to learn a unified classifier over incrementally occurring sets of classes. Since all the incremental data cannot be retained for unified training, the major challenge is to avoid forgetting previous classes while learning new ones.

The three crucial components of a class-IL algorithm include a memory buffer to store few exemplars from old classes, a forgetting constraint to keep previous knowledge while learning new tasks, and a learning system that balances old and new classes. Although several methods have been proposed to address each of these components, there is not yet a common understanding of best practices.

Prabhu *et al*. [22] provides an overview over current state of continual learning methods for classification. It shows that a simple greedy balanced sampler-based approach (GDumb) can outperform various specialized formulations in most of the continual learning settings, however, it finds class-IL particularly challenging. In this work, we propose a complementary approach to [22] for class-IL, where softmax outputs are masked appropriately with data balancing to outperform previous sophisticated approaches.

Contributions. We propose a compositional class-IL (CCIL) model that isolates the underlying reasons for catastrophic forgetting in class-IL and combines the most simple and most effective components to build a robust base model. It employs plain knowledge distillation [11] as a forgetting constraint and selects exemplar samples simply randomly. For the loss evaluation, we propose important changes in the output normalization. The goal of this part (Section 3 & 4) is to show that a balanced usage of simple components is sufficient to produce a strong model with state-of-the-art performance.

In addition, we study the influence of the learned representation's properties on forgetting and show that the degree of feature specialization (overfitting) correlates with the degree of forgetting. We study some common regularization techniques and show that only those that keep, or even improve, the so-called secondary class information – also referred as dark knowledge by [11] – have a positive influence on class-incremental learning, whereas others make things much worse. The source code of this paper is available [1].

2. Related Work

iCaRL was the first approach that formally introduced the class-IL problem [23]. iCaRL is a decoupled approach

[1]Source code: https://github.com/sud0301/essentials_for_CIL

for feature representation learning and classifier learning. It alleviates catastrophic forgetting via knowledge distillation and a replay-based approach. Later Castro *et al*. [3] extended it to an end-to-end learning model based on a combination of distillation and cross-entropy loss to show improved results over iCaRL. Successive works usually dedicated their contribution to one of the three components in class-IL.

Exemplar selection: Replay-based approaches have been shown to be quite effective in mitigating catastrophic forgetting. Typically, a memory buffer is allocated to store exemplar samples of old classes, which are replayed while learning a new task to mitigate forgetting. Many works [3, 12, 23, 29] use herding heuristics [28] for exemplar selection. Herding selects and retains samples closest to the mean sample for each class. Liu *et al*. [17] parameterized the exemplars to optimize them jointly with the model. Iscen *et al*. [13] introduced a memory efficient approach to store feature descriptors instead of images. In our work, we simply sample from each class randomly to compile the exemplar set.

Forgetting-constraint: Knowledge distillation (KD) was first introduced by Li *et al*. [16] for multi-task incremental learning. Thereafter, various works [3, 23, 29] have adopted it in class-IL to restore previous knowledge. Lately, several works have proposed new forgetting constraints with an objective to preserve the structure of old-class embeddings. Hou *et al*. [12] proposed the usage of feature-level distillation by penalizing change is the feature representation from the old model. Yu *et al*. [31] utilized an embedding network to rectify the semantic drift, Tao *et al*. [25] proposed a Hebbian graph-based approach to retain the topology of the feature space. In this work, we utilize plain knowledge distillation, which is based on logits to avoid forgetting.

Bias removal methods: Various works [12, 29, 33] have pointed out that class-imbalance between old and new classes creates a bias in the class weight vectors in the last linear layer, due to which the network predictions are biased towards new classes. To rectify this bias, Wu *et al*. [29] trained an extra bias-correction layer using the validation set, Belouadah *et al*. [2] proposed to rectify the final activations using the statistics of the old task predictions, Zhao *et al*. [33] adjusted the norm of new class-weight vectors to those of the old class-weight vectors, and Hou *et al*. [12] applied cosine normalization in the last layer. The focus of these works is limited to the bias in the last layer, but ultimately catastrophic forgetting is an issue that affects the entire network: class imbalance causes the model to overfit to the new task, deteriorating the performance on the old ones. Some works [3, 15] also fine-tune the model to avoid overfitting to the current task. We propose a learning system that resolves this bias without the need of any post-processing, by fixing the underlying issues; see Section 4.

3. Class-Incremental Learning

3.1. Problem Definition

The objective of class-incremental learning (class-IL) is to learn a unified classifier from a sequence of data from different classes. Data arrives incrementally as a batch of per-class sets $\mathcal{X}$ *i.e.* $(X^1, X^2, ..., X^t)$, where X^y contains all images from class y. Learning from a batch of classes can be considered as a task $\mathcal{T}$. At each incremental step, the data for the new task $\mathcal{T}_i$ arrives, which contains samples of the new set of classes. At each step, complete data is only available for new classes $\mathcal{X}$ *i.e.* $(X^{s+1}, ..., X^t)$. Only a small amount of exemplar data $\mathcal{P}_{old}$ *i.e.* $(P^1, ..., P^s)$ from previous classes *i.e.* $(X^1, ..., X^s)$ is retained in a memory buffer of limited size. The model is expected to classify all the classes seen so far.

The problem definition with strictly separated batches may appear a bit specific. In many practical applications, the data will arrive in a more mixed-up fashion. However, this strict protocol allows the comparison of techniques and it covers the key issues with class-incremental learning.

3.2. Evaluation Metrics for Class-IL

Class-IL models are evaluated using three metrics: average incremental accuracy, forgetting rate and feature retention. After each incremental step, all classes seen so far are evaluated using the latest model. After N incremental tasks, the accuracy $\mathcal{A}_n$ over all $(N+1)$ steps is averaged and reported. It is termed as *average incremental accuracy* (**Avg Acc**), introduced by Rebuffi *et al*. [23]. We also evaluate the *forgetting rate* $\boldsymbol{\mathcal{F}}$ proposed by Liu *et al*. [17]. The forgetting rate measures the performance drop on the first task. It is the accuracy difference on the classes of the first task $X^{1:s}_{test}$, using Θ_0 and Θ_N. Therefore, it is independent of the absolute performance on the initial task $\mathcal{T}_0$. We introduce another metric, referred as $\boldsymbol{\mathcal{R}}_\phi$, to measure retention in the feature extractor $\phi(\cdot)$. It measures how much information is retained in the feature extractor while learning the tasks incrementally as compared to a jointly trained model. To measure $\boldsymbol{\mathcal{R}}_\phi$: after the final incremental step, parameters of the feature extractor are frozen and the last linear layer is learned using all the data from all the classes. $\boldsymbol{\mathcal{R}}_\phi$ is the accuracy difference between this model and a model where the whole network is trained on all the classes with complete data access.

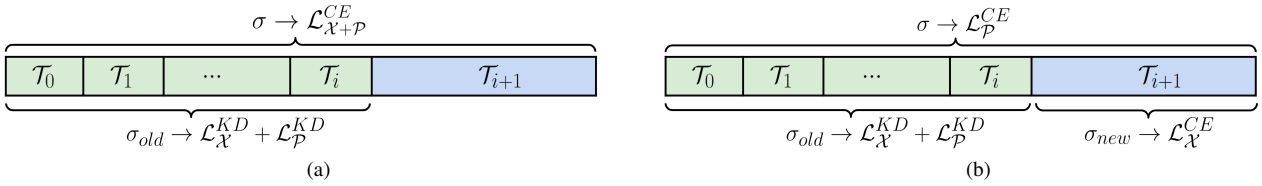


Figure 1: The comparison between a (a) standard loss system and our proposed (b) compositional loss system (right). σ shows the softmax function span over all the network output logits. σ_{old} and σ_{new} shows softmax span over the set of old and new class logits respectively.

3.3. A Basic Class-IL Framework

The network model Θ consists of a feature extractor $\phi(\cdot)$ and a fully-connected layer $fc(\cdot)$ for classification. Similar to a standard multi-class classifier, the output logits o are processed through a softmax activation function $\sigma(\cdot)$ before cross-entropy loss $\mathcal{L}^{CE}$ is evaluated corresponding to the correct class. For the initial base task $\mathcal{T}_0$, the model Θ^s learns a standard classifier for the first ($y \in y[1:s]$) classes. In the incremental step, the fc layer is adapted to learn new classes ($y \in y[s+1:t]$) by adding new output nodes, whereas the other part of the network remains unchanged, resulting into a new model Θ^t. The three main elements of class-IL are set up as follows.

Exemplar selection: We compile the exemplar set by randomly selecting an equal number of samples (m) for each class. The samples are sorted in ascending order according to the distance from the mean of the feature vectors μ_i for each class separately. Since the size of the limited memory is fixed (K), some samples of old classes are removed to accommodate exemplars from new classes. Samples with larger distances to the mean vector are removed first. Detailed steps are shown in Algorithm 2.

Forgetting constraint: Our model uses knowledge distillation as the forgetting constraint. Knowledge distillation penalizes the change with respect to the output of the old model (Θ^s) using KL-divergence, thus preserving the network's knowledge about the old classes. The distillation loss ($\mathcal{L}^{KD}$) is computed for the exemplar sets ($\mathcal{P}$) as well as for samples from the new classes ($\mathcal{X}$). The final loss for our CCIL model is a combination of cross-entropy loss $\mathcal{L}^{CE}$ for classification and distillation loss $\mathcal{L}^{KD}$ for mitigating catastrophic forgetting as shown in Algorithm 1-Line 15.

Learning system: We propose a new compositional learning system which addresses the weight-bias issue in class-IL. The proposed loss isolates inter-task and intra-task learning for a balanced processing of data by appropriately normalizing the output logits. The task-agnostic parts are shared to yield improved efficiency. The details are presented in the next section.

4. Compositional Learning System

For each gradient update, the CCIL model receives data in separate mini-batches from the set of new classes $\mathcal{X}$ and the set of exemplars $\mathcal{P}$. $\mathcal{P}$ is the updated exemplar set which also includes equal size of exemplars from the current new classes (see Algorithm 1-Line 3). The losses on set $\mathcal{X}$ and $\mathcal{P}$ are computed as:

$$\mathcal{L}_{\mathcal{X}} = \mathcal{L}^{CE}_{\mathcal{X}} + \lambda * \mathcal{L}^{KD}_{\mathcal{X}} \quad (1)$$

$$\mathcal{L}_{\mathcal{P}} = \mathcal{L}^{CE}_{\mathcal{P}} + \lambda * \mathcal{L}^{KD}_{\mathcal{P}} \quad (2)$$

Intra-task Learning: The classification loss for the new classes ($\mathcal{L}^{CE}_{\mathcal{X}}$) is computed using a dedicated softmax function σ_{new} comprising logits of new classes only (Figure 1b)

Algorithm 1: CCIL: IncrementalStep

Input: $\mathcal{X} = (X^{s+1}, ..., X^t), \mathcal{P}^s = (P_1, ..., P_s)$ // new classes data, old exemplar sets
Input: $K, \Theta^s, \hat{\Theta}^s$ // memory size, current model, frozen current model
Output: Θ^t // model trained on t classes

1 $m \leftarrow K/t$ // number of exemplars per class
2 $\Theta^t \leftarrow \Theta^s$ // add output nodes for new classes
3 $\mathcal{P} \leftarrow$ UpdateExemplarSets$(\mathcal{X}; \mathcal{P}^s, m, \Theta^s)$
4 **for** $(x, y) \in \mathcal{X}$ **do** //
update for mini-batch data in $\mathcal{X}$
5 $o = \Theta^t(x)$ // $o = \{o_{old}, o_{new}\}$
6 softmax over new class logits $\sigma_{new}(o_{new})$
7 compute classification loss $\mathcal{L}^{CE}_{\mathcal{X}}$ (Eq. 3)
8 softmax over old class logits $\sigma_{old}(o_{old})$
9 compute distillation loss $\mathcal{L}^{KD}_{\mathcal{X}}$ (Eq. 4)
10 **load a mini-batch from exemplars set** $(x', y') \sim \mathcal{P}$
11 $o' = \Theta^t(x')$
12 softmax over all logits $\sigma(o')$
13 compute classification loss $\mathcal{L}^{CE}_{\mathcal{P}}$ (Eq. 5)
14 compute distillation loss $\mathcal{L}^{KD}_{\mathcal{P}}$ (Eq. 6)
15 $\mathcal{L} = (\mathcal{L}^{CE}_{\mathcal{X}} + \mathcal{L}^{CE}_{\mathcal{P}}) + \lambda * (\mathcal{L}^{KD}_{\mathcal{X}} + \mathcal{L}^{KD}_{\mathcal{P}})$
16 **end**

computed as:

$$\mathcal{L}_{\mathcal{X}}^{CE} = -\sum_{i=s+1}^{t} y[i] \cdot \log(p_{new}[i]) \tag{3}$$

for $(x, y) \in \mathcal{X}$, where $p_{new} = \sigma_{new}(o_{new})$, $o = \Theta^t(x)$ and output logits comprise $o = \{o_{old}, o_{new}\}$. This allows the classifier weights for the new classes to be learned independently of the previous classes – while sharing the feature extractor, thus effectively eliminating the weight bias. Distillation loss ($\mathcal{L}_{\mathcal{X}}^{KD}$) is always computed using σ_{old} (see Figure 1b), since output of new network $p_{old} = \sigma_{old}(o_{old})$ are compared against the output of previous model $\hat{p} = \sigma_{old}(\hat{\Theta}^s(x))$ as:

$$\mathcal{L}_{\mathcal{X}}^{KD} = D_{KL}(\hat{p}||p_{old}) \tag{4}$$

In case of a unified softmax, the weights of the old classes are suppressed by the larger amount of new class samples during training. A similar intra-task learning method using separate-softmax has been concurrently proposed in [1].

Inter-task Learning: The separate softmax helps intra-task learning for the new classes, but this does not yet discriminate the new from the old classes. For inter-task learning, we plan a balanced interaction between the samples of old and new classes. We compile an exemplar set $\mathcal{P}$ which contains equal numbers of samples from all classes including old and new classes. However small, such exemplar set enables the model to capture the inter-task relationship through the loss $\mathcal{L}_{\mathcal{P}}^{CE}$, which uses a combined softmax function σ evaluated on all classes (see Figure 1b).

$$\mathcal{L}_{\mathcal{P}}^{CE} = -\sum_{i=1}^{t} y'[i] \cdot \log(q[i]) \tag{5}$$

for $(x', y') \in \mathcal{P}$, where $q = \sigma(o')$ and $o' = \Theta^t(x')$. The distillation loss is computed similar to Eq. 4,

$$\mathcal{L}_{\mathcal{P}}^{KD} = D_{KL}(\hat{q}||q_{old}) \tag{6}$$

where $\hat{q} = \sigma_{old}(\hat{\Theta}^s(x'))$ and $q_{old} = \sigma_{old}(o'_{old})$. This exemplar set is compiled before learning the incremental task, contrary to previous works, where it is always compiled after the incremental step. Figure 1 shows how the loss terms are calculated using a separate softmax function 1b and also compares it to the unified softmax 1a used in previous works.

Transfer Learning: We observed that a separate softmax does not remove the bias completely. Another cause for unbalanced class-weight vectors, and catastrophic forgetting in general, is the change in the data distribution between different tasks. We hypothesize that the effect of this distribution shift in the training data is more harmful to the previous knowledge when the transfer learning from old to new classes is poor, resulting in strong alteration of the parameters of the network. We propose to reduce the learning rate for the incremental steps as a simple way to improve transfer learning and mitigate the adverse effect of distribution shift. This further helps reduce the weight bias. The reduced learning rate on incremental steps depends on the scale and relevance of features learned in the base task, therefore it is determined experimentally. Although lowering the learning rate is a standard technique when fine-tuning a network on a new dataset, its importance is underestimated and often missing in incremental learning works. Section 6.2 contains ablation studies to show its importance.

Algorithm 2: UpdateExemplarSets

Input: $\mathcal{X}, \mathcal{P}_{old}$ // new class data, old exemplar set
Input: Θ^s, m // old model, new exemplar size per class
Output: $\mathcal{P}_{new}$ // new Exemplar sets

```
for i = 1, ..., s do
    P_i ← (p_1, ..., p_m) // keep first m samples
end
/* add new class exemplars */
for i = s + 1, ..., t do
    P_i ← (p_1, ..., p_m) ⊂ X^i) // randomly pick m samples
    μ_i ← (1/m) Σ_{j=1}^{m} φ(p_j) // mean feature
    /* sort exemplars based on distance from μ_i */
    for k = 1, ..., m do
        p_k ← arg min ||μ_i − φ(p_k)||
    end
end
```

5. Improving Feature Representations for Incremental Learning

Intuitively, poorly transferable embeddings will force the model to alter its parameters significantly in order to learn new concepts. This destroys the knowledge accumulated for the previous tasks. In this section, we explore this novel direction – aiming to learn robust representations that are transferable to a new task and effectively retain previous knowledge in class-IL. In particular, we study the detrimental effects of overfitting and loss of secondary class information. We find that: 1) both phenomena strongly correlate with catastrophic forgetting; 2) regularization methods can significantly improve robustness against forgetting, but only as long as they enhance the secondary class information of the learned model.

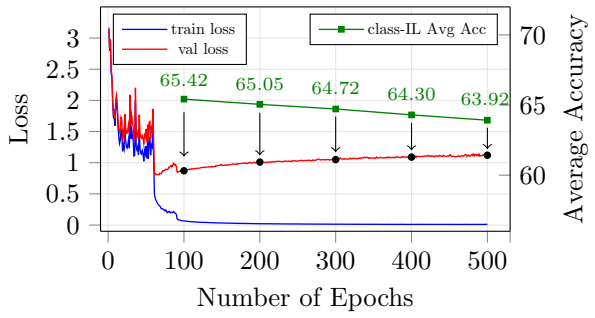


Figure 2: The effect of overfitting on class-IL performance on the CIFAR-100 dataset. Figure shows the overfitting behavior on the initial base task. The validation loss (red curve) starts increasing monotonically after the 100^{th} epoch. The green curve shows the average incremental accuracy (right y-axis) for class-IL experiments performed over different snapshots at every 100^{th} epoch.

Epoch	SS-NLL ↓	SS-Acc ↑	$\mathcal{F}\downarrow$	$\mathcal{R}_\phi\downarrow$
100	2.54	38.68	16.03	9.04
200	2.89	32.88	16.04	9.27
300	3.03	30.09	16.94	9.51
400	3.09	29.04	18.38	9.68
500	3.11	27.97	18.57	10.00

Table 1: The effect of overfitting on class-IL performance and its correlation with secondary information, on the CIFAR-100 dataset. Table shows the performance of the snapshots taken at every 100^{th} epoch and the corresponding class-IL model. SS-Acc decreases and SS-NLL increases as more overfitted models are evaluated. Forgetting rate $\mathcal{F}$ and feature retention metric $\mathcal{R}_\phi$ also correlate with overfitting. Results are averaged over 5 runs, standard deviation is reported in Appendix.

5.1. Measuring the Quality of Secondary Logits

Secondary information captures semantic relationship between the target and non-target classes. In literature, the term *secondary information* is interchangeably used to denote the non-target and non-maximum scores of a classifier [30]. Here, for evaluation purposes, the term denotes the non-maximum scores produced by the networks. When applying the maximum operation to the scores predicted by a classifier, part of the information produced by the model is discarded. For each individual sample this information represents the model's belief about the semantic nature of the image, in relation to the other classes. It is important to learn this secondary information, such that the model can re-use it to learn new classes with least modification to previous concepts. We argue that semantically similar classes should lie closer in the representation space as compared to the dissimilar classes since they share more features, and higher secondary information is an indicator of such an efficient non-redundant feature space. Appendix includes an analysis on feature representations to support this argument.

No proper annotations exist for secondary information, therefore we define a proxy evaluation objective, exploiting the *coarse*-labeling of the CIFAR-100 dataset, which partitions the 100 *fine*-classes into 20 superclasses. The 5 classes belonging to each superclass are mostly semantically related, and have been previously used for evaluating secondary information [30]. As a proxy evaluation measure for secondary class information we propose to use the classification performance on the superclasses, restricting the network output to the non-maximum logits. We define two new metrics for this purpose: Secondary Superclass NLL and Secondary Superclass Accuracy.

Secondary Superclass-NLL (SS-NLL): Negative Log Likelihood is a commonly used cost function for classification, also known as *Cross-Entropy Loss*. Here we compute the NLL induced by the secondary (non-maximum) logits on the superclass classification problem. Given a set of superclasses $\mathcal{S}$, we can group the fine-grained classes into subsets $\mathcal{C}$ according to their coarse-label, and compute:

$$SS\text{-}NLL(x,y) = -\sum_{j\in\mathcal{S}}\left[\mathbb{1}_{\mathcal{C}_j}(y)\log\sum_{k\in\mathcal{C}_j}\hat{\sigma}_k\big(f(x)\big)\right], \tag{7}$$

where $\mathbb{1}_{\mathcal{C}_j}(y)$ is an indicator function which evaluates to 1 if the true class y belongs to superclass j, $\hat{\sigma}$ is a softmax function over the secondary fine-logits (i.e. it suppresses the maximum logit). The network prediction (logits) is denoted as $f(x)$. A lower SS-NLL indicates better superclass classification, thus higher secondary information quality.

Secondary Superclass-Accuracy (SS-Acc): Secondary superclass accuracy computes the percentage of correct superclass predictions. As for SS-NLL, the largest logit score is excluded from the prediction to focus the measure on the quality of secondary information. Higher SS-Acc values indicate higher quality of the secondary information.

5.2. Forgetting starts before the incremental step

In this section, we study how the quality of the representations learned during the initial base task correlates with incremental learning performance. We experimentally show how a decline in quality of the learned features–measured as overfitting and loss of secondary information – leads to higher catastrophic forgetting, motivating our following search for a suitable regularizer.

Experiment details: We set up a class-IL experiment (with 5 incremental tasks) on CIFAR-100.The initial base network is trained for up to 500 epochs. We employ a SGD

Model	Avg. Acc.↑		SS Metrics		Forgetting	F. Retention	ECE↓
	5 tasks	10 tasks	SS-NLL ↓	SS-Acc. ↑	$\mathcal{F}\downarrow$	$\mathcal{R}_\phi\downarrow$	
CCIL	66.44	64.86	2.784	34.83	17.13	9.70	0.100
CCIL + SD	67.17	65.86	2.675	37.26	16.81	8.88	0.094
CCIL + H-Aug	71.66	69.88	2.051	47.69	13.37	6.73	0.018
CCIL + LS	63.08	61.99	3.103	24.25	18.79	12.83	0.049
CCIL + Mixup	62.31	57.75	2.791	31.57	24.56	16.01	0.024

Table 2: Effect of regularization class-IL average accuracy, secondary information (on the first-task model), forgetting rate and feature retention (5 tasks), on CIFAR-100. All the values are averaged over 3 runs. ↓ and ↑ in the column headings indicate that lower and higher values are better respectively. Values that are better than the CCIL baseline are marked in green whereas the worse ones are marked in red. SD:self-distillation, LS:label-smoothing, H-Aug:heavy data augmentation. Standard deviation in Appendix A.

optimizer with a base learning of 1e-1. We use a step learning rate schedule, where the learning rate is divided by 10 at 60^{th} and 90^{th} epochs.

Analysis: Figure 2 shows that the validation loss (red curve) starts increasing after about 100 epochs, showing an overfitting effect. Thereafter, we perform five different class-IL experiments, each based on a different snapshot of the base network (every 100^{th} epoch). As the validation loss of the snapshot increases, incremental learning performance of the corresponding class-IL model drops (green curve), and both forgetting rate ($\mathcal{F}$) and feature retention metric ($\mathcal{R}_\phi$) worsen (Table 1). The worsening $\mathcal{R}_\phi$ metric indicates that the issue is rooted in the feature representations, and cannot be mitigated by acting on the last layer bias. Along with these metrics, we observe that overfitting causes the quality of secondary information to deteriorate (SS-Acc decreases and the SS-NLL increases, Table 1). This loss of secondary information could also be linked to increasing overconfidence of the network, measured as Expected Calibration Error (ECE) [9] (details in Appendix A).

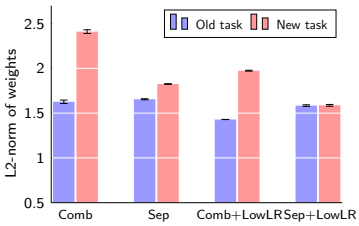
These results indicate that: 1) the quality of the features learned during the first base task influences the performance of the class-IL model, and as such it should be expressly addressed. 2) secondary information can be considered as an indicator of the features' quality and their fitness for incremental learning. In the next section we will show experimental evidence in support of these hypotheses.

5.3. Analyzing Catastrophic Forgetting with Regularization

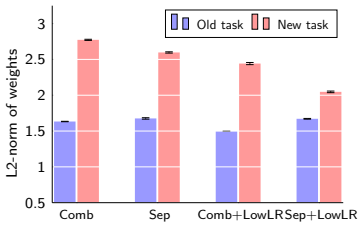
Having established a link between early feature quality and catastrophic forgetting, we hypothesize that the application of adequate regularization techniques can improve model performance on the task at hand. We apply four common regularization techniques to our CCIL model: self-distillation [7], data-augmentation (including cropping, cutout [6] and an extended set of AutoAugment [4] policies), label smoothing [24], and mixup [32]. All these regularizers have been shown to improve generalization on the held-out validation data. We report details about the application of said regularization methods in Appendix A.

Self-distillation [7, 20] is a form of knowledge distillation in which the teacher and student networks have the same architecture. It can be applied iteratively, in generations: at each generation a copy of the current student becomes the new teacher. **Data Augmentation** is one of the most widespread regularization techniques for neural networks, especially in computer vision. A well designed data augmentation routine is key to obtaining good results on the held-out dataset. We sample randomly from a pool of augmentation policies which contain pairs of different geometric and color transformations, similarly to [4]. **Label smoothing** [24] acts on the cross-entropy loss for classification by interpolating the one-hot labels with a uniform distribution over the possible classes. This technique has been shown to improve generalization and reducing overconfidence of classification models [24]. **Mixup** [32] is an operation that generates training samples for classification by linearly combining pairs of existing samples – image and label. Mixup has successfully been used as a form of data augmentation in image classification, improving generalization and calibration [32, 26].

Analysis We analyse above discussed metrics for each of these regularization techniques. Table 2 shows the Average Accuracy after finishing the last incremental step, secondary information quality of the first task model, forgetting rate, feature retention (Section 3.2) and expected calibration error [9]. We can divide the regularization methods into two groups: the ones which improve class-IL performance (self-distillation, augmentation) and the ones which harm it (label smoothing, mixup). The first group also shows consistent improvements in secondary information and reduction in forgetting, with augmentation performing the best across all metrics – by a significant margin. In the second group, label smoothing harms secondary information the most. It has been observed that label smoothing encourages repre-



(a)



(b)

Operations	Avg Acc ↑ w/o KD	Avg Acc ↑ w/ KD
Comb	47.97	52.71
Sep	52.86	60.85
Comb+LowLR	52.79	54.54
Sep+LowLR	**58.60**	**64.79**

(c)

Figure 3: (a) & (b) compares the average L_2 norm of the classification weight vectors for old and new classes for class-IL experiments without (w/o) and with (w/) KD respectively. We evaluate standard combined softmax (Comb) against proposed separate softmax (Sep) and we assess the effect of reduced learning rate (LowLR). (c) contains the corresponding class-IL results without distillation (w/o KD) and with distillation (w/ KD) in terms of average incremental accuracy. Figure shows how bias is reduced using separate softmax and reduced learning rate. All experiments use the linear classification layer. Results shown on CIFAR-100 for 5-task experiments.

sentations to be closer to their respective class centroid and equidistant to the other class centroids [21], and this comes at the expense of inter-class sample relationships, i.e., secondary information. Mixup also harms the quality of secondary information: we believe this is because it artificially forces arbitrary distances between classes, which modifies the natural output distribution – similarly to label smoothing. Interestingly, all regularizers improve network calibration, but ECE is not a good indicator of class-IL performance, unlike secondary information, shown in Table 2.

In summary, label smoothing and mixup – despite their proven regularization effects – harm secondary class information and have clear negative consequences for class-incremental learning. On the other hand, regularization methods that enhance secondary class information (self distillation and data augmentation) boost the average incremental accuracy. Analogously to the analysis of Section 5.2 we show that the quality of secondary information negatively correlates to the forgetting rate (Table 2), further indicating the importance of secondary class information.

6. Results

6.1. Training Details

Datasets We conduct experiments on CIFAR100 [14], ImageNet-100 Subset [5] and full ImageNet datasets. The ImageNet-100 dataset has 100 randomly sampled classes (using Numpy seed:1993) from ImageNet. The base CCIL model uses default data augmentation including random cropping and horizontal flipping for CIFAR-100, and resized-random cropping and horizontal flipping for ImageNet datasets. All the randomization seeds are selected following the experiments in previous works [12, 17].

Benchmark protocol We follow the protocol used in previous works [12, 17]. The protocol involves learning of 1 initial base task followed by N incremental tasks. We evaluate with two incremental settings: where the model learns $N = 5$ and $N = 10$ incremental tasks. For CIFAR-100 and ImageNet-100, 50 classes are selected as the base classes for the initial task and the remaining classes are equally divided over the incremental steps and for ImageNet, 500 base classes are used. Exemplar memory size is set to $K = 2k$ for 100 class datasets and $K = 20k$ for ImageNet.

Implementation details We use a 32-layer ResNet [10] for CIFAR-100 dataset, and a 18-layer ResNet for ImageNet-100 and ImageNet datasets. The last layer is cosine normalized following the recommendations of [12].

6.2. Ablation Studies

Elements of the compositional learning system We evaluate the contributions of each element in the proposed learning system by training multiple class-IL models featuring them. The incremental learning in these experiments is conducted in two settings – in a simple fine-tuning setup (without distillation), in order to single out the effects of the proposed changes and with distillation loss. In Figure 3a & 3b we compare the average L2 norm of the class weight vectors for old and new classes after 5 incremental steps, while in Figure 3c we provide the average accuracies of the respective models. We notice a major difference in the weight norms of old and new classes for the default combined softmax (Comb) setting (Figure 1a). Using separate-softmax (Sep) substantially reduces this difference and improves class-IL performance, but does not resolve the problem completely. Lower learning rate (Comb+LowLR) also reduces the bias and improves the performance, although to a lesser extent. When both approaches are combined (Sep+Low-LR), this bias is further reduced and the best class-IL results are produced. We observe a marginal difference in the last norms in Figure 3b because the distillation loss can only be applied using the old class logits and does not use the compositional learning system (Sec. 4). Compared to [1], our proposed inter-task learning module increases the performance by 1% on CIFAR-100 dataset over

Method	Layer		Softmax		Low LR	AW	Classifier		KD	Avg Acc
	Cos	Dot	Sep	Comb			NME	CNN		
Comb		✓		✓				✓		47.97
iCaRL		✓		✓			✓		✓	56.50
iCaRL++	✓			✓		✓		✓	✓	59.78
CCIL	✓		✓		✓	✓		✓	✓	66.44

Table 3: Drawing parallels between iCaRL and our proposed model. Average accuracy is reported for 5-task class-IL experiments on CIFAR-100 dataset. Last row highlights our proposed changes. All methods use random exemplar selection as used in this work, Dot: linear layer, KD: knowledge distillation, NME: nearest-mean-of-exemplars (used in [23])

Method	CIFAR-100		ImageNet-100		ImageNet	
No. of incremental tasks →	5	10	5	10	5	10
iCaRL* [23]	57.17	52.57	65.04	59.53	51.50	46.89
BIC [29]	59.36	54.20	70.07	64.96	62.65	58.72
WA [33]	63.25	58.57	—	—	—	—
LUCIR [12]	63.12	60.14	70.47	68.09	64.34	61.28
Mnemonics [17]	63.34	62.28	72.58	71.37	64.54	63.01
TPCIL [25]	65.34	63.58	76.27	74.81	64.89	62.88
CCIL (ours)	66.44	64.86	77.99	75.99	67.53	65.61
CCIL-SD (ours)	**67.17**	**65.86**	**79.44**	**76.77**	**68.04**	**66.25**
Joint-training	74.12	73.80	84.72	84.67	69.72	69.75

Table 4: Comparing average accuracy using different methods on CIFAR-100, ImageNet-100 and ImageNet dataset. *reported in [12]

only using intra-task learning module as proposed in [1]-v1.

Drawing parallels with iCaRL We compare different components of our CCIL model with the first baseline approach (iCaRL) proposed by [23]. Table 3 summarizes these changes. We first isolate the contributions of some follow-up methods by creating another baseline as iCaRL++. It consists of a (1) cosine-normalized layer (cos) [8, 18, 12], where the features and class-weight vectors in the final layer are normalized to lie in a high-dimensional sphere. It helps in removing the remaining weight bias during inference, and (2) adaptive weighting (AW), where the weight of the distillation loss increases with incremental steps. AW was previously introduced in [12], which helps in adaptive balancing of classification and distillation loss (more details are included in the Appendix). The last row shows that replacing the combined-softmax (comb) with the proposed separate-softmax (sep) and reducing the learning rate (LowLR) yields a major improvement.

6.3. Comparison to SOTA

Results for CIFAR-100, ImageNet-100 and ImageNet datasets are shown in Table 4. We report the upper bound 'Joint-training', where at every incremental step all the data for the classes seen until then is accessible. The simple CCIL model compares favorably to previous results on all datasets, especially on larger datasets like ImageNet-1k. The regularized CCIL-SD closes the gap to joint training further and achieves state-of-the-art performance across all datasets. Since the CCIL model is based only on simple components, the application of advanced methods for mitigating forgetting [12, 25] and more informative exemplar selection [17] can further improve the performance.

7. Conclusions

We presented a straightforward class-incremental learning system that focuses on the essential components and already exceeds the state of the art without integrating sophisticated modules. This makes it a good base model for future research on advancing class-incremental learning.

Moreover, we showed that countering catastrophic forgetting during the incremental step is not enough: the quality of the feature representation prior to the incremental step considerably determines the amount of forgetting. In this regard we showed that boosting secondary information is key to improve the transferability of features from old to new tasks without forgetting. We believe this discovery is generic to all continual learning settings and is a promising direction for future work.

Acknowledgements

This study was supported by the German Federal Ministry of Education and Research via the project Deep-PTL.

References

[1] Hongjoon Ahn, Jihwan Kwak, Subin Lim, Hyeonsu Bang, Hyojun Kim, and Taesup Moon. Ss-il: Separated softmax for incremental learning. *arXiv preprint arXiv:2003.13947*, 2020. 4, 7, 8

[2] Eden Belouadah and Adrian Popescu. Il2m: Class incremental learning with dual memory. In *IEEE International Conference on Computer Vision*, 2019. 2

[3] Francisco M. Castro, Manuel J. Marin-Jimenez, Nicolas Guil, Cordelia Schmid, and Karteek Alahari. End-to-end incremental learning. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018. 2

[4] Ekin D. Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V. Le. Autoaugment: Learning augmentation strategies from data. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2019. 6

[5] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2009. 7

[6] Terrance DeVries and Graham W Taylor. Improved regularization of convolutional neural networks with cutout. *arXiv preprint arXiv:1708.04552*, 2017. 6

[7] Tommaso Furlanello, Zachary Chase Lipton, Michael Tschannen, Laurent Itti, and Anima Anandkumar. Born-again neural networks. In *International Conference on Machine Learning*, 2018. 6

[8] Spyros Gidaris and Nikos Komodakis. Dynamic few-shot visual learning without forgetting. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2018. 8

[9] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. On calibration of modern neural networks. In *Proceedings of the 34th International Conference on Machine Learning*, 2017. 6

[10] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2016. 7

[11] Geoffrey Hinton, Oriol Vinyals, and Jeffrey Dean. Distilling the knowledge in a neural network. In *NIPS Deep Learning and Representation Learning Workshop*, 2015. 1

[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*, 2019. 2, 7, 8

[13] Ahmet Iscen, Jeffrey Zhang, Svetlana Lazebnik, and Cordelia Schmid. Memory-efficient incremental learning through feature adaptation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020. 2

[14] Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, 2009. 7

[15] Kibok Lee, Kimin Lee, Jinwoo Shin, and Honglak Lee. Overcoming catastrophic forgetting with unlabeled data in the wild. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019. 2

[16] Z. Li and D. Hoiem. Learning without forgetting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2018. 2

[17] Yaoyao Liu, Yuting Su, An-An Liu, Bernt Schiele, and Qianru Sun. Mnemonics training: Multi-class incremental learning without forgetting. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2020. 2, 7, 8

[18] Chunjie Luo, Jianfeng Zhan, Xiaohe Xue, Lei Wang, Rui Ren, and Qiang Yang. Cosine normalization: Using cosine similarity instead of dot product in neural networks. In *International Conference on Artificial Neural Networks*, 2018. 8

[19] Michael Mccloskey and Neil J. Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. *The Psychology of Learning and Motivation*, 1989. 1

[20] Hossein Mobahi, Mehrdad Farajtabar, and Peter L. Bartlett. Self-distillation amplifies regularization in hilbert space. *arXiv:12002.05715*, 2020. 6

[21] Rafael Müller, Simon Kornblith, and Geoffrey E Hinton. When does label smoothing help? In *Advances in Neural Information Processing Systems*, 2019. 7

[22] Ameya Prabhu, Philip H. S. Torr, and Puneet K. Dokania. Gdumb: A simple approach that questions our progress in continual learning. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020. 1

[23] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H. Lampert. icarl: Incremental classifier and representation learning. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2017. 1, 2, 8

[24] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2016. 6

[25] Xiaoyu Tao, Xinyuan Chang, Xiaopeng Hong, Xing Wei, and Yihong Gong. Topology-preserving class-incremental learning. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020. 2, 8

[26] Sunil Thulasidasan, Gopinath Chennupati, Jeff A. Bilmes, Tanmoy Bhattacharya, and Sarah Michalak. On mixup training: Improved calibration and predictive uncertainty for deep neural networks. In *Advances in Neural Information Processing Systems*, 2019. 6

[27] Gido M. van de Ven and Andreas S. Tolias. Three scenarios for continual learning. *arXiv:1904.07734*, 2019. 1

[28] Max Welling. Herding dynamical weights to learn. In *International Conference on Machine Learning*, 2009. 2

[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*, 2019. 2, 8

[30] Chenglin Yang, Lingxi Xie, Siyuan Qiao, and Alan L. Yuille. Knowledge distillation in generations: More tolerant teachers educate better students. In *Association for the Advancement of Artificial Intelligence*, 2019. 5

[31] Lu Yu, Bartlomiej Twardowski, Xialei Liu, Luis Herranz, Kai Wang, Yongmei Cheng, Shangling Jui, and Joost van de

Weijer. Semantic drift compensation for class-incremental learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 2

[32] Hongyi Zhang, Moustapha Cisse, Yann N. Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. In *International Conference on Learning Representations*, 2018. 6

[33] Bowen Zhao, Xi Xiao, Guojun Gan, Bin Zhang, and Shu-Tao Xia. Maintaining discrimination and fairness in class incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 2, 8