HCV: Hierarchy-Consistency Verification for Incremental Implicitly-Refined Classification

Kai Wang[1]
kwang@cvc.uab.es
Xialei Liu (*corresponding author)[2]
xialei@nankai.edu.cn
Luis Herranz[1]
lherranz@cvc.uab.es
Joost van de Weijer[1]
joost@cvc.uab.es

[1] Computer Vision Center
Universitat Autònoma de Barcelona
Barcelona, Spain

[2] College of Computer Science
Nankai University
Tianjin, China

Abstract

Human beings learn and accumulate hierarchical knowledge over their lifetime. This knowledge is associated with previous concepts for consolidation and hierarchical construction. However, current incremental learning methods lack the ability to build a concept hierarchy by associating new concepts to old ones. A more realistic setting tackling this problem is referred to as Incremental Implicitly-Refined Classification (IIRC), which simulates the recognition process from coarse-grained categories to fine-grained categories. To overcome forgetting in this benchmark, we propose Hierarchy-Consistency Verification (HCV) as an enhancement to existing continual learning methods. Our method incrementally discovers the hierarchical relations between classes. We then show how this knowledge can be exploited during both training and inference. Experiments on three setups of varying difficulty demonstrate that our HCV module improves performance of existing continual learning methods under this IIRC setting by a large margin. Code is available in https://github.com/wangkai930418/HCV_IIRC.

1 Introduction

In the lifetime of a human being, knowledge is continuously learned and accumulated. However, deep learning models suffer from knowledge forgetting, also known as catastrophic forgetting [11, 21], when presented with a sequence of tasks. Incremental learning [5, 19, 24], also referred to as continual learning, has been a crucial research direction in computer vision that aims to prevent this forgetting of previous knowledge in neural networks.

Another aspect of human learning is the association of new concepts to old concepts, people construct a hierarchy of knowledge to better consolidate this information. Recently, the IIRC (Incremental Implicitly-Refined Classification) setup [1] has been proposed as a novel extended benchmark to evaluate lifelong learning methods in a realistic setting where the construction of hierarchical knowledge is key. On the IIRC benchmark (see Fig. 1), each class has multiple granularity levels. But only one label is present at any time, which requires the model to infer whether the related labels have been observed in previous tasks. This

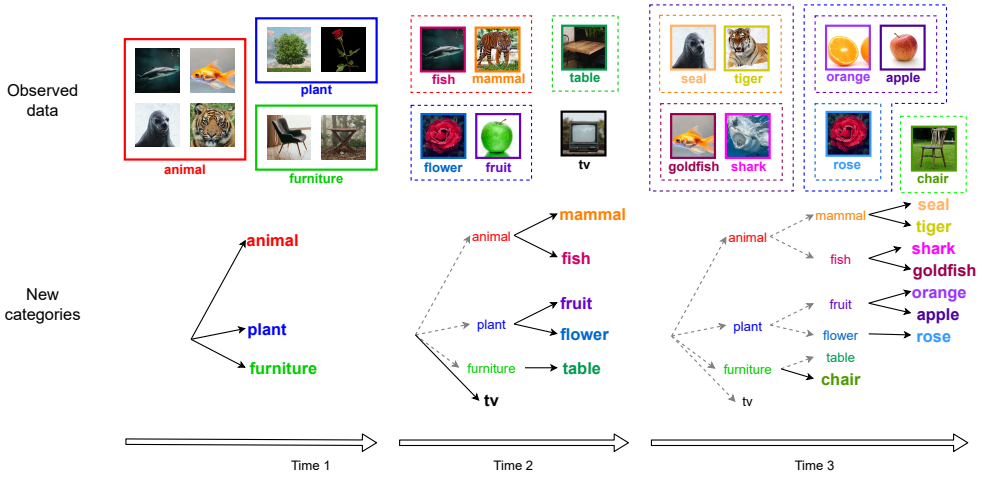


Figure 1: Illustration of 3-layer hierarchy IIRC setting. New categories in each training time are annotated by solid pointers, and the hierarchical relationships among old categories and new categories are denoted with dashed arrows.

setting is much closer to real-life learning, where a learner gradually improves its knowledge of objects (first it labels roses as a plant, later as a flower, and finally a rose).

Based on this benchmark, Abdelsalam et al. [1] adapted and evaluated several state-of-the-art incremental learning methods to address this problem, including iCaRL [27], LUCIR [9], and AGEM [4]. However, their work does not propose an effective solution specifically designed for the IIRC problem. They do not aim to incrementally learn the hierarchical knowledge that is important to correctly label the data in this setting. Furthermore, there are also some other limitations in the current version of the IIRC benchmark: (i) The granularity is limited to two layers, while in reality there are often more layers involved (see WordNet [22] hierarchy of ImageNet [6]). (ii) The first task always contains a large number of superclasses, which means that the learner encounters data from most classes already in these early stages[1]. This makes training relatively easy, and the proposed setup less applicable.

To overcome catastrophic forgetting under the IIRC setup, we propose a module called Hierarchy-Consistency Verification (HCV). We aim to explicitly learn in an incremental manner the hierarchical knowledge that underlies the data. While learning new tasks with new super and subclasses, we automatically discover relations, e.g. the class 'flower' is a subclass of 'plant'. Next, we show how this knowledge can be exploited to enhance incremental learning. Principally, in the described example, we would not use images from 'flower' as negative examples for the class 'plant' (a problem from which the methods in [1] suffer). Next, we show how the hierarchical knowledge can be used at inference time to improve the predictions. Based on these observations, our main contributions are:

- We propose a Hierarchy-Consistency Verification (HCV) module as a solution to the IIRC setup. It incrementally discovers the hierarchical knowledge underlying the data, and exploits this during both training and inference.

[1]The actual setup considers 10 superclasses in the first task, meaning that around 50 (of the total 100) subclasses are seen implicitly during the first task.

- We extend the IIRC benchmark to a challenging 3-layer hierarchy on the IIRC-CIFAR dataset. In addition, we propose a much harder setup where the superclasses are distributed uniformly over incremental tasks to test the robustness of different methods.
- Experiments show that we successfully acquire hierarchical knowledge, and that exploiting this knowledge leads to significantly improvements of existing incremental learning methods under the IIRC setup (with absolute accuracy gains of 3-20%).

2 Related work

2.1 Incremental learning

Incremental learning methods can be categorized into three types [6, 19] as follows.

Regularization-based methods. The first group of techniques add a regularization term to the loss function which impedes changes to the parameters deemed relevant to previous tasks. The difference depends on how to compute the estimation. These methods can be further divided into data-focused [10, 14, 26, 34] and prior-focused [2, 8, 11, 13, 15, 33]. Data-focused methods use knowledge distillation from previously learned models. Prior-focused methods estimate the importance of model parameters as a prior for the new model.

Parameter isolation methods. This family focuses on allocating different model parameters to each task. These models begin with a simplified architecture and updated incrementally with new neurons or network layers in order to allocate additional capacity for new tasks. In Piggyback/PackNet [17, 18], the model learns a separate mask on the weights for each task, whereas in HAT [28] masks are applied to the activations. This method is further developed to the case where no forgetting is allowed in [20]. In general, this branch is restricted to the task-aware (task incremental) setting. Thus, they are more suitable for learning a long sequence of tasks when a task oracle is present.

Replay methods. This type of methods prevent forgetting by including data from previous tasks, stored either in an episodic memory or via a generative model. There are two main strategies: exemplar rehearsal [4, 9, 16, 27, 32] and pseudo-rehearsal [29, 31]. The former stores a small amount of training samples (also called exemplars) from previous tasks. The latter use generative models learned from previous data distributions to synthesize data.

2.2 Hierarchical classification and multi-label classification

Classification problem is normally considered that the categories are not overlapped with each other. However, the concepts in real life are connected to each other with hierarchical information. For example, in ImageNet [6], the categories are hierarchized by WordNet [22] knowledge. For hierarchical classification [30], the system groups things according to an explicit hierarchy, which is important to some applications, such as bioinformatics [7] and COVID-19 identification [25]. Another related area is multi-label classification [35], where each image is related to multiple labels. Multi-label classification is a generalization of the single-label categorizing problem. In the multi-label problem there is no constraint on how many of the classes the instance can be assigned to. While under this setup, there is no hierarchical constraints among categories. By comparison, on the IIRC setup [1], the hierarchical information is implicitly defined. The developed model for this problem should be able to learn this hierarchy by itself and predict the multiple labels for each instance.

3 Methodology

The original work that presented the IIRC setup [1] ignores the hierarchical nature of the classes during incremental learning. Consequently, some samples are incorrectly used as negative samples for their superclass labels, potentially resulting in a drop of performance. Here we propose our method to incrementally learn the hierarchy and directly exploit this information to remove said interference. Moreover, we also show how the estimated hierarchy can be exploited at inference time. Our method is general and can be applied to existing methods for incremental learning that can be trained with a binary cross-entropy loss (in experiments we will show results for iCaRL [27], and LUCIR [9]).

3.1 IIRC setup

Given a series of tasks, each task $t \in [1,T]$ is composed of data D_t from the current class set C_t which can contain both super- and subclasses. During training of task t the model will receive $(x_t^i, y_t^i) \in D_t^{train}$, $y_t^i \in C_t$ where $y_t^i \in \{u_t^i, v_t^i\}$ is either the subclass u_t^i or the superclass v_t^i label of the i-th sample x_t^i, only one of which is present in C_t. In the proposed setup of [1], always first the superclass is learned and later the subclass (like in Fig. 1). We will use lowercase y for a one-hot vector, and capital Y to identify a binary vector possibly with multiple non-zero elements. It is important to note that even if during training only a single label y_t^i is provided, during testing after task t we consider test data $(x_t^i, Y_t^i) \in \cup_{j=1}^{t} D_j^{test}$ where multi-class ground-truth vector Y_t^i contains the subclass and superclass label of sample x_t^i (if these are in $\cup_1^t C_t$[2]), i.e., at test time we are expected to predict all non-zero elements in Y_t^i.

To make the common recognition model applied in this multi-class case, in [1] they propose to replace the conventional cross-entropy loss by a binary cross-entropy loss:

$$\mathcal{L}_{BCE} = -\sum_i [y_t^i \cdot log(\hat{Y}_t^i) + (1 - y_t^i) \cdot log(1 - \hat{Y}_t^i)] \tag{1}$$

where $\hat{Y}_t^i = \mathcal{F}_t(x_t^i)$ is the predicted probability vector of sample x_t^i, with $\mathcal{F}_t$ the current prediction model. They apply this equation to several incremental learning algorithms. However, it should be noted that samples can be wrongly used as a negative sample for their own superclass, because this loss only considers the provided label y_t^i.

We extend the two-layer hierarchy proposed in the original IIRC setup to three layers to verify the effectiveness of our module in more complex scenarios. In this case, each sample contains a three-layer label annotations Y_t^i as: (subclass u_t^i, superclass v_t^i, rootclass w_t^i).

3.2 HCV: Hierarchy-Consistency Verification

In the previous section, we discussed that the original solution results in interference during training. The challenge here is that the model should correctly learn the relationship between sub classes u_t^i and super classes v_t^i, given only the y_t^i information during training time. Here, we propose our method that address this problem.

To overcome forgetting under the IIRC setup, we incrementally compute the class hierarchy by estimating the relationship between old and new classes. If a new class is highly related to an old class, we identify it as the subclass of the old class. With this estimated

[2] Some samples might only have a single label since the subclass label is not yet encountered during training.

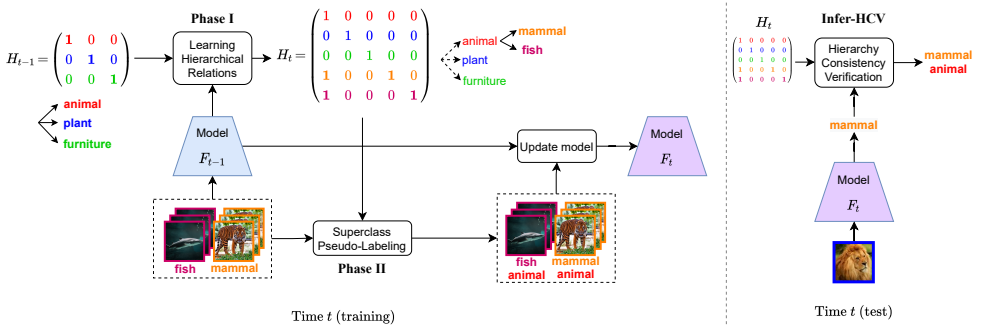


Figure 2: Illustration of our method: Hierarchy-Consistency Verification (HCV). At Phase I, hierarchical relations between subclasses and superclasses H_t are acquired using current data. And then at Phase II, the multi-class labels are generated for each instance. Current model is updated with calibrated labels at training time. The hierarchical relations can be applied during inference time as well to further improve the predictions.

hierarchical knowledge, we verify the hierarchy consistency both during training and inference time to boost the performance of the continual learning models. Our algorithm, called *Hierarchy-Consistency Verification* (HCV), contains two phases which we describe in the following (see also Fig. 2). Moreover, the learned hierarchy is also exploited at inference.

Phase I: Learning Hierarchical Relations (LHR). The mission at this stage is to estimate the existing hierarchical relationship between subclasses u_t^i and superclasses v_t^i. This stage occurs before the training of the current task. Supposing we have learned the classifier $\mathcal{F}_{t-1}$ for all previous classes. We could use $\mathcal{F}_{t-1}$ to classify all accessible data D_t^{train} for class y_c and produce a prediction vector p_{y_c}.

$$p_{y_c} = \frac{1}{N} \cdot \sum_{i|y_t^i=y_c} \mathcal{F}_{t-1}(x_t^i) \quad (x_t^i, y_t^i) \in D_t^{train} \tag{2}$$

where N is the number of images labeled as $y_c \in C_t$. If the maximum prediction value in p_{y_c} is larger than a threshold τ, we would consider the previous class $\bar{v}_t^i$ with the max probability value is the superclass of class y_t^i. Based on this prior knowledge learned from previous classifiers, we could construct a hierarchical tree H_t, which consists of all hierarchical information up to the current task t.

Phase II: Superclass Pseudo-Labeling (SPL). After learning the superclasses before training task t, we have the hierarchical tree H_t, which contains all estimated hierarchical information up to the current task. Now we can apply this knowledge at both train and test time.

During training time, if a new class is estimated as a subclass of a specific previous superclass, we assign the estimated superclass label $\bar{v}_t^i$ as a *superclass pseudo-label* to the corresponding subclasses label y_t^i (we will use the overline $\bar{\cdot}$ to identify that label is estimated). In this way, the estimated multi-class label $\bar{Y}_t^i$ can be represent as:

$$\bar{Y}_t^i = \begin{cases} y_t^i & \text{if } y_t^i \text{ has no parents in the hierarchical tree } H_t \\ y_t^i \cup \bar{v}_t^i & \text{if } \bar{v}_t^i \text{ is the estimated parent of } y_t^i \end{cases} \tag{3}$$

Then, with the new class label vector $\bar{Y}_t^i$, the binary cross-entropy loss is rewritten as:

$$\mathcal{L}_{BCE} = -\sum_i [\bar{Y}_t^i \cdot log(\hat{Y}_t^i) + (1-\bar{Y}_t^i) \cdot log(1-\hat{Y}_t^i)] \tag{4}$$

For applying our SPL module to continual learning methods, we simply replacing the orginal BCE loss in Eq. 1 with Eq. 4.

Inference with HCV (Infer-HCV). At inference time, if a multi-class prediction vector is not consistent with our estimated hierarchical knowledge H, we mark it as a wrong prediction (e.g. it estimates a sub and superclass combination that is not in accordance to our hierarchical knowledge captured by H). Based on this assumption, we process each prediction $\hat{Y}_t^i$ with H_t. If the prediction is in accordance with H_t it remains unchanged. If we need to add labels to $\hat{Y}_t^i$ to make it be in accordance to H_t we do so (add subclass or superclass label). If we need to remove labels from $\hat{Y}_t^i$ to reach accordance with H_t, we randomly select one of the possible solutions containing the least number of removed labels. See the supplementary material for a visual explanation of Infer-HCV.

4 Experiments

4.1 Experimental setup

Datasets. We use the same two datasets as in IIRC [1]: CIFAR100 [12] and ImageNet [6]. For CIFAR100, we take the two-level hierarchy split IIRC-CIFAR from IIRC [1], we denote this as IIRC-2-CIFAR. It is composed of 15 superclasses and 100 subclasses. To further explore the performance of incremental learning methods over multi-level hierarchy, we further extend the IIRC-2-CIFAR into a three-level hierarchy dataset IIRC-3-CIFAR with two highest superclasses (we name them as "root"): "animals" and "plants". That accounts 2 rootclasses, 15 superclasses and 100 subclasses. For ImageNet, due to its huge amount of data, we collect 100 subclasses according to the hierarchy proposed in IIRC [1]. In total there are 10 superclasses and 100 subclasses (including those have no superclass labels). We denote this dataset as IIRC-ImageNet-Subset as a simplified version of the original one. The detailed hierarchies and task information are referred to the supplementary material.

Incremental task configurations. For IIRC-2-CIFAR, we adopt the training sequence from IIRC [1], where the first task is with 10 superclasses, in the sequential tasks each with 5 classes. And for IIRC-3-CIFAR, we uniformly distribute the rootclasses and superclasses to form 23 tasks in total, the first task is 7 classes and then the coming tasks are 5 for each. For IIRC-ImageNet-Subset, we have 11 tasks each with 10 classes. Here the superclasses are also uniformly distributed. We want to stress that the uniform distribution of superclasses (and rootclasses) leads to a more challenging setting than proposed in the original IIRC.

Baselines and Compared methods. We compare the performance of the following variants: (1) **Incremental Joint** learns the model across tasks and the model has access to all the data from previous tasks with complete information (having access to all the label annotations Y_t). It serves as the upper bound for comparison. (2) **ER-infinite** is similar to *Incremental Joint* but with incomplete information (only access to the current label annotations y_t). (3) **iCaRL-CNN** is the original version of incremental learning method iCaRL [27]. (4) **iCaRL-norm** is the adapted version of iCaRL [27] with replacement of the distance metric

Methods	iCaRL-CNN			iCaRL-norm			LUCIR		
	-	+ SPL	+ SPL + infer HCV	-	+ SPL	+ SPL + infer HCV	-	+ SPL	+ SPL + infer HCV
IIRC-2-CIFAR	28.4	32.7	**35.9**	24.9	29.1	31.9	28.5	33.0	34.7
IIRC-3-CIFAR	20.5	26.0	27.1	19.6	25.6	25.9	16.1	35.5	**37.2**
IIRC-ImageNet-Subset	28.7	29.3	**31.7**	28.2	29.1	31.3	23.3	26.8	28.2

Table 1: We show the average of *pw-JS* from comparison over three datasets with and without our HCV module. + *SPL* means applying HCV in training stage, + *Infer-HCV* means applying HCV module in inference time.

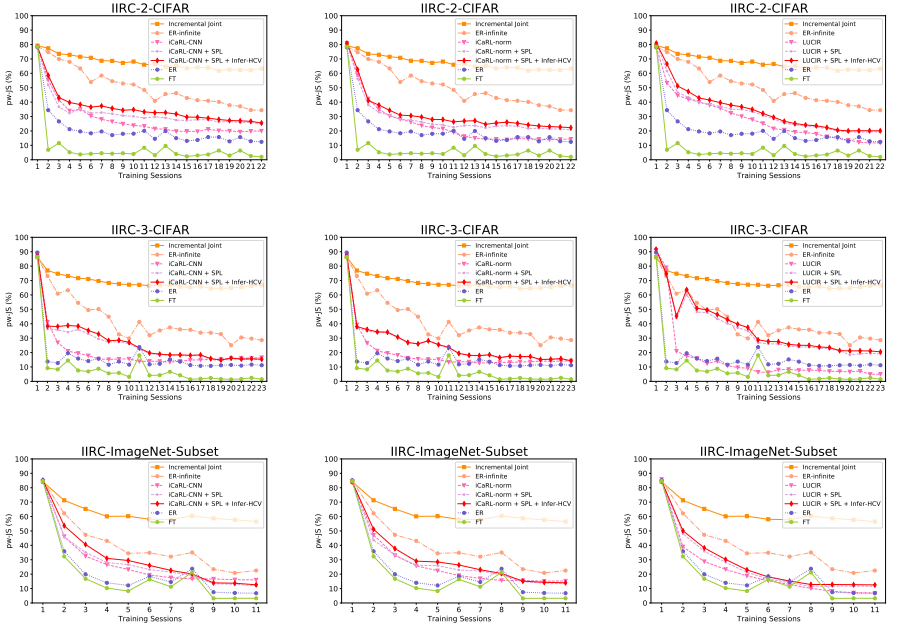
from L2-distance to Cosine similarity. (5) **LUCIR** is the incremental learning method LUCIR [9]. (6) **ER** is the finetuning baseline with 20 image exemplars per class as experience replay. (7) **FT** is the finetuning baseline without image replay.

Implementation details. For most implementation details, we follow the IIRC configurations [1]. For these three setups, we use the ResNet-32 [8] as the classification backbone. For model training, we use SGD (momentum=0.9) as optimizer, which is commonly used in continual learning [2, 3]. For the IIRC-2-CIFAR and IIRC-3-CIFAR setting, the learning rates begin with 1.0 then decay by 0.1 on the plateau of the validation performance. For IIRC-ImageNet-Subset, the learning rate starts with 0.5 and decay by 0.1 on the plateau. The number of training epochs is 140, 140 and 100 for IIRC-2-CIFAR, IIRC-3-CIFAR and IIRC-ImageNet-Subset, respectively. For all these three setups, the batch size is 128 and weight decay is 1e-5.

During training, we apply random resized cropping (of size 32×32) to both CIFAR100 and ImageNet images. Then a random horizontal flip is applied and followed by a normalization. And for images replay, we keep a fixed number of 20 saved exemplars per class by default. For evaluation, we adopt the *precision-weighted Jaccard similarity (pw-JS)* proposed in IIRC [1], which integratedly considers both precision and recall indexes. And the threshold τ is set to 0.6 in all experiments (except in ablation study over it).

4.2 Experimental results

HCV applied to existing methods. To verify the performance of our proposed HCV, we apply it to iCaRL-CNN, iCaRL-norm and LUCIR. The average *pw-JS* value is provided in Table 1. We conduct experiments using three different settings, that is IIRC-2-CIFAR, IIRC-3-CIFAR and IIRC-ImageNet-Subset. On IIRC-2-CIFAR setting, with the help of our HCV module during the training stage, the average numbers are increased by nearly 4.3% for all three different continual learning methods. When we apply HCV also at inference time, it further improves the consistency of final predictions achieving the average number by 3.2%, 2.8%, 1.7% for these three methods respectively. On the IIRC-3-CIFAR setting, since it is a much harder setup for incremental learning, all these variants suffer a significant drop of performance. LUCIR is much better compared to iCaRL-CNN and iCaRL-norm. Applying HCV in both training and inference stages helps to boost performance around 6.5% for two iCaRL variants and 21.1% for LUCIR. IIRC-ImageNet-Subset setting has much higher image diversity, thus it also imposes difficulties for these incremental methods. Under this setting, LUCIR performs worse than iCaRL-CNN and iCaRL-norm even with the improvement from HCV. And iCaRL-CNN works similar to iCaRL-norm but with marginally better performance. Overall, using our proposed HCV during training and inference improves performance of existing methods consistently for different settings.



(a) iCaRL-CNN (b) iCaRL-norm (c) LUCIR

Figure 3: Experimental results over IIRC-2-CIFAR, IIRC-3-CIFAR and IIRC-ImageNet-Subset setups based on three methods: iCaRL-CNN, iCaRL-norm and LUCIR.

Final estimated hierarchy graph and visual examples. After learning the last task under IIRC-2-CIFAR setup when applying our SPL module to iCaRL-CNN, we estimate the full hierarchy and draw a subgraph with 3 superclasses in Fig. 4 (right). We can observe that most subclasses are correctly annotated with its superclasses. However *table* is not correctly annotated because its confidence (58%) does not reach the threshold. Interestingly, *television* is wrongly classified as a subclass of *furniture*. In real life, we could also regard it as a member of *furniture* and this was learned because *televisions* occur often in *furniture* scenes. This kind of information can help human operators in annotating and verifying the dataset hierarchy. Further, we see that *house, bridge, castle* are false positives, and are classified as subclasses of *vehicles*. This could be because *vehicles* images co-occur with the *house, bridge, castle* classes as their background. Finally, we also show some visual examples from IIRC-2-CIFAR setup and in-the-wild images in Fig. 4(left).

Comparison with SOTA methods. In Fig. 3 we plot the dynamic performance changes of different methods. The general trend on different settings are similar. Incremental Joint always achieves the best results as an upper bound, benefiting from access to all data and labels, while ER-infinite lacks the knowledge of full labels resulting in a worse performance. Our proposed HCV improves existing methods consistently, but the gap between our best and the two upper bounds (ER-infinite and Incremental Joint) is still large, which shows that IIRC setting is a very challenging setting requiring more research.

Confusion matrices. Fig. 5 shows the confusion matrices after learning task 11 under IIRC-2-CIFAR setup. They are from the ground truth, original continual learning methods, and HCV applied to both training and inference time. It can be observed that after using HCV, the redundant predictions are cleaned with our learned prior knowledge about the classes

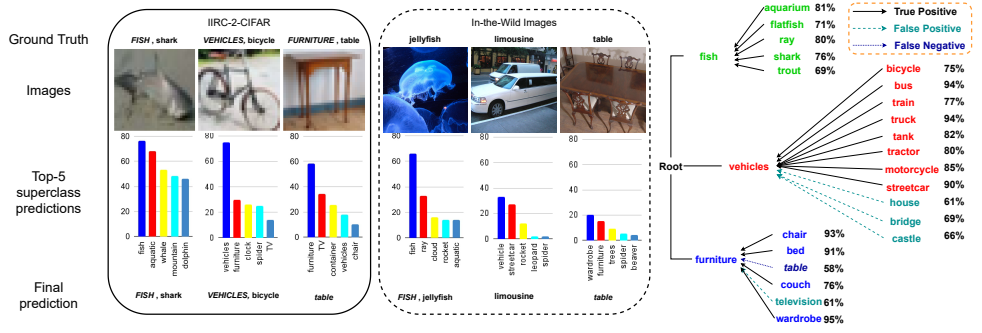
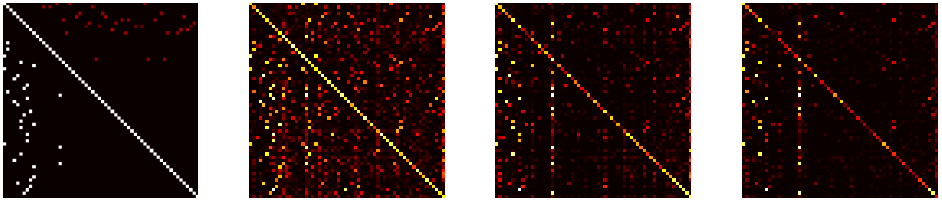


Figure 4: Visual examples of our model applied to IIRC-2-CIFAR setup (annotated with superclasses and subclasses) and in-the-wild images (annotated with class names). We plot the top-5 (ranked by % percentage) predicted superclasses for each query image. We take the default threshold $\tau = 0.6$ to distinguish the success and failure cases. A subgraph of the final predicted graph under IIRC-2-CIFAR setup with iCaRL method is shown on the right. Here the top-1 predicted superclasses with percentages are listed.



(a) Ground truth (b) iCaRL-CNN (c) + SPL (d) +SPL+HCV

Figure 5: Confusion matrices of groundtruth, original continual learning methods, applying SPL and applying Infer-HCV to iCaRL-CNN after task 11 under IIRC-2-CIFAR setup.

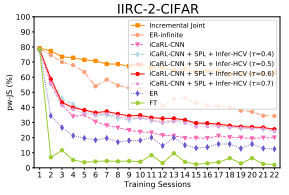
hierarchy, therefore HCV plays a role of a de-noising procedure for confusion matrices.

4.3 Ablation study

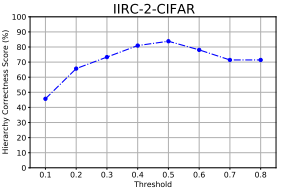
Ablation study over threshold τ. We conduct an ablation study on the threshold τ under IIRC-2-CIFAR setup. In Fig. 6a, we compare the values of τ {0.4, 0.5, 0.6, 0.7} when applying HCV on both training and inference stages. We can observe that with different hyper-parameters, it improves over iCaRL-CNN consistently. In Fig. 6b, we show how the hierarchy correctness score (HCS) changes with the threshold from 0.1 to 0.8, and is around 75% to 80% when τ is in the range [0.3, 0.7]. In our experiments, we set $\tau = 0.6$ by default.

Ablation study over hierarchy correctness score (HCS). We also conduct an ablation study over the HCS on LUCIR and ER methods as shown in Fig. 6d and Fig. 6e. The hierarchy correctness scores for iCaRL, LUCIR, ER are 76.2%, 56.0%, 34.3%, respectively (the HCS curves by training sessions are shown in Fig. 6c). The higher hierarchy correctness score for iCaRL-CNN helps it achieve state-of-the-art performance on IIRC-2-CIFAR and IIRC-ImageNet-Subset (Table 1 and Fig. 3). While LUCIR achieves a much lower score though it is regarded as one of the best methods in continual learning [19].

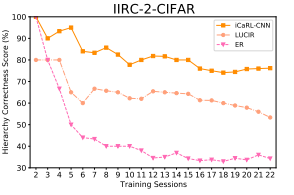
We also show the performance of the LUCIR and ER methods with the ground-truth hierarchy, which means it has a HCS of 100% (see Fig. 6d and Fig. 6e). In this case 3.0%



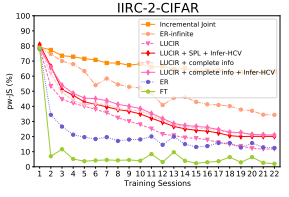
(a) Ablation on threshold τ



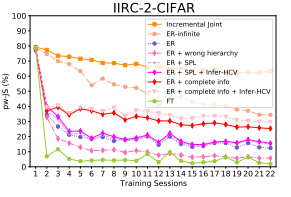
(b) HCS changes by τ



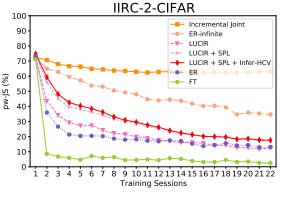
(c) HCS curves of 3 methods



(d) Ablate HCS on LUCIR



(e) Ablate HCS on ER



(f) LUCIR with 10 orders

Figure 6: Ablation study over threshold τ, HCS and class orders on IIRC-2-CIFAR setup.

and 15.0% improvements are observed for LUCIR and ER respectively. That implies that our HCV module can benefit from a preciser hierarchy estimation to reduce the gap to ER-infinite. To test how a completely wrong class hierarchy influences our model, we randomly generate a hierarchy for IIRC-2-CIFAR and apply it to ER (Fig. 6e), we can observe a drop of HCS from 34.3% to 0.0%, and the overall performance drops for ER to nearly 7.0%.

HCV (on LUCIR) performance with 10 orders. In Fig. 6f the experiments are conducted with all 10 task-orderings proposed in IIRC [1]. We plot the average performance. Here we apply our SPL and Infer-HCV to the LUCIR model. We observe a significant and consistent improvement compared to the ER baseline (≈10.0%) and the basic LUCIR method (≈8.0%). In conclusion, our method improves the performance under various orders and settings.

5 Conclusion

In this paper, we proposed a Hierarchy-Consistency Verification module for Incremental Implicit-Refined Classification (IIRC) problem. With this module, we can boost the existing incremental learning methods by a large margin. From our experiments on three different setups, we evaluate and prove the effectiveness of our proposed module during both training and inference. And from the visualization of confusion matrices, we can also find that our HCV module works as a denoising method to the confusion matrices. For future work, we are interested in associating hierarchical classification, multi-label classification with IIRC problem, thus to have a more robust model to overcome forgetting in more realistic setups.

Acknowledgement

We acknowledge the support from Huawei Kirin Solution, the Spanish Government funding for projects PID2019-104174GB-I00 and RTI2018-102285-A-I00, and Kai Wang acknowledges the Chinese Scholarship Council (CSC) No.201706170035. Herranz acknowledges the Ramón y Cajal fellowship RYC2019-027020-I.

References

[1] Mohamed Abdelsalam, Mojtaba Faramarzi, Shagun Sodhani, and Sarath Chandar. Iirc: Incremental implicitly-refined classification. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2021.

[2] Rahaf Aljundi, Francesca Babiloni, Mohamed Elhoseiny, Marcus Rohrbach, and Tinne Tuytelaars. Memory aware synapses: Learning what (not) to forget. In *European Conference on Computer Vision*, pages 139–154, 2018.

[3] Arslan Chaudhry, Puneet K Dokania, Thalaiyasingam Ajanthan, and Philip HS Torr. Riemannian walk for incremental learning: Understanding forgetting and intransigence. In *European Conference on Computer Vision*, pages 532–547, 2018.

[4] Arslan Chaudhry, Marc'Aurelio Ranzato, Marcus Rohrbach, and Mohamed Elhoseiny. Efficient lifelong learning with a-gem. In *International Conference on Learning Representations*, 2019.

[5] Matthias Delange, Rahaf Aljundi, Marc Masana, Sarah Parisot, Xu Jia, Ales Leonardis, Greg Slabaugh, and Tinne Tuytelaars. A continual learning survey: Defying forgetting in classification tasks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.

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

[7] Alex Freitas and André Carvalho. A tutorial on hierarchical classification with applications in bioinformatics. *Research and trends in data mining technologies and applications*, pages 175–208, 2007.

[8] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

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

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

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

[12] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[13] Sang-Woo Lee, Jin-Hwa Kim, Jaehyun Jun, Jung-Woo Ha, and Byoung-Tak Zhang. Overcoming catastrophic forgetting by incremental moment matching. *Advances in Neural Information Processing Systems*, 2017.

[14] Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(12):2935–2947, 2017.

[15] Xialei Liu, Marc Masana, Luis Herranz, Joost Van de Weijer, Antonio M Lopez, and Andrew D Bagdanov. Rotate your networks: Better weight consolidation and less catastrophic forgetting. In *Proceedings of the International Conference on Pattern Recognition*, pages 2262–2268. IEEE, 2018.

[16] Xialei Liu, Chenshen Wu, Mikel Menta, Luis Herranz, Bogdan Raducanu, Andrew D Bagdanov, Shangling Jui, and Joost van de Weijer. Generative feature replay for class-incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 226–227, 2020.

[17] Arun Mallya and Svetlana Lazebnik. Packnet: Adding multiple tasks to a single network by iterative pruning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7765–7773, 2018.

[18] Arun Mallya, Dillon Davis, and Svetlana Lazebnik. Piggyback: Adapting a single network to multiple tasks by learning to mask weights. In *European Conference on Computer Vision*, pages 67–82, 2018.

[19] Marc Masana, Xialei Liu, Bartlomiej Twardowski, Mikel Menta, Andrew D Bagdanov, and Joost van de Weijer. Class-incremental learning: survey and performance evaluation. *arXiv preprint arXiv:2010.15277*, 2020.

[20] Marc Masana, Tinne Tuytelaars, and Joost van de Weijer. Ternary feature masks: continual learning without any forgetting. *2nd CLVISION workshop in CVPR 2021*, 2020.

[21] Michael McCloskey and Neal J Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of learning and motivation*, volume 24, pages 109–165. Elsevier, 1989.

[22] George A Miller. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41, 1995.

[23] Seyed Iman Mirzadeh, Mehrdad Farajtabar, Razvan Pascanu, and Hassan Ghasemzadeh. Understanding the role of training regimes in continual learning. *arXiv preprint arXiv:2006.06958*, 2020.

[24] German I Parisi, Ronald Kemker, Jose L Part, Christopher Kanan, and Stefan Wermter. Continual lifelong learning with neural networks: A review. *Neural Networks*, 113: 54–71, 2019.

[25] Rodolfo M Pereira, Diego Bertolini, Lucas O Teixeira, Carlos N Silla Jr, and Yandre MG Costa. Covid-19 identification in chest x-ray images on flat and hierarchical classification scenarios. *Computer Methods and Programs in Biomedicine*, 194: 105532, 2020.

[26] Amal Rannen, Rahaf Aljundi, Matthew B Blaschko, and Tinne Tuytelaars. Encoder based lifelong learning. In *Proceedings of the International Conference on Computer Vision*, pages 1320–1328, 2017.

[27] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. icarl: Incremental classifier and representation learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2001–2010, 2017.

[28] Joan Serra, Didac Suris, Marius Miron, and Alexandros Karatzoglou. Overcoming catastrophic forgetting with hard attention to the task. In *International Conference on Machine Learning*, pages 4548–4557. PMLR, 2018.

[29] Hanul Shin, Jung Kwon Lee, Jaehong Kim, and Jiwon Kim. Continual learning with deep generative replay. In *Advances in Neural Information Processing Systems*, 2017.

[30] Carlos N Silla and Alex A Freitas. A survey of hierarchical classification across different application domains. *Data Mining and Knowledge Discovery*, 22(1):31–72, 2011.

[31] Chenshen Wu, Luis Herranz, Xialei Liu, Joost van de Weijer, Bogdan Raducanu, et al. Memory replay gans: Learning to generate new categories without forgetting. In *Advances in Neural Information Processing Systems*, pages 5962–5972, 2018.

[32] Yue Wu, Yinpeng Chen, Lijuan Wang, Yuancheng Ye, Zicheng Liu, Yandong Guo, and Yun Fu. Large scale incremental learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 374–382, 2019.

[33] Friedemann Zenke, Ben Poole, and Surya Ganguli. Continual learning through synaptic intelligence. In *International Conference on Machine Learning*, pages 3987–3995. PMLR, 2017.

[34] Junting Zhang, Jie Zhang, Shalini Ghosh, Dawei Li, Serafettin Tasci, Larry Heck, Heming Zhang, and C-C Jay Kuo. Class-incremental learning via deep model consolidation. In *Proceedings of the IEEE Workshop on Applications of Computer Vision*, pages 1131–1140, 2020.

[35] Min-Ling Zhang and Zhi-Hua Zhou. A review on multi-label learning algorithms. *IEEE transactions on knowledge and data engineering*, 26(8):1819–1837, 2013.