LwM

Learning without Memorizing

Prithviraj Dhar*[1], Rajat Vikram Singh*[2], Kuan-Chuan Peng[2], Ziyan Wu[2], Rama Chellappa[1]
[1]University of Maryland, College Park, MD
[2]Siemens Corporate Technology, Princeton, NJ
{prithvi,rama}@umiacs.umd.edu, {singh.rajat,kuanchuan.peng,ziyan.wu}@siemens.com

Abstract

Incremental learning (IL) is an important task aimed at increasing the capability of a trained model, in terms of the number of classes recognizable by the model. The key problem in this task is the requirement of storing data (e.g. images) associated with existing classes, while teaching the classifier to learn new classes. However, this is impractical as it increases the memory requirement at every incremental step, which makes it impossible to implement IL algorithms on edge devices with limited memory. Hence, we propose a novel approach, called 'Learning without Memorizing (LwM)', to preserve the information about existing (base) classes, without storing any of their data, while making the classifier progressively learn the new classes. In LwM, we present an information preserving penalty: Attention Distillation Loss (L_{AD}), and demonstrate that penalizing the changes in classifiers' attention maps helps to retain information of the base classes, as new classes are added. We show that adding L_{AD} to the distillation loss which is an existing information preserving loss consistently outperforms the state-of-the-art performance in the iILSVRC-small and iCIFAR-100 datasets in terms of the overall accuracy of base and incrementally learned classes.

1. Introduction

Most state-of-the-art solutions to visual recognition tasks use models that are specifically trained for these tasks [6, 13]. For the tasks involving categories (such as object classification, segmentation), the complexity of the task (i.e. the number of target classes) limits the ability of these trained models. For example, a trained model aimed for object recognition can only classify object categories on which it has been trained. However, if the number of target classes increases, the model must be updated in such a way that it performs well on the original classes on which it has been trained, also known as base classes, while it incrementally learns new classes as well.

If we retrain the model only on new, previously unseen classes, it would completely forget the base classes, which is known as catastrophic forgetting [9, 10], a phenomenon which is not typically observed in humane learning. Therefore, most existing solutions [4, 14, 18] explore incremental learning (IL) by allowing the model to retain a fraction of the training data of base classes, while incrementally learning new classes. Yu et al. [18] proposed retaining trained models encoding base class information, to transfer their knowledge to the model learning new classes. However, this process is not scalable. This is because storing base class data or models encoding base class information is a memory expensive task, and hence is cumbersome when used in a lifelong learning setting. Also, in an industrial setting, when a trained object classification model is delivered to the end user, the training data is kept private for proprietary reasons. Therefore, the end user will be unable to update the trained model to incorporate new target classes in the absence of base class data.

Moreover, storing base class data for incrementally learning new classes is not biologically inspired. For example, when a toddler learns to recognize new shapes/objects, it is observed that it does not completely forget the shapes or objects it already knows. It also does not always need to revisit the old information when learning

*These authors have contributed equally to this work, which was partially done during PD's internship at Siemens Corporate Technology.

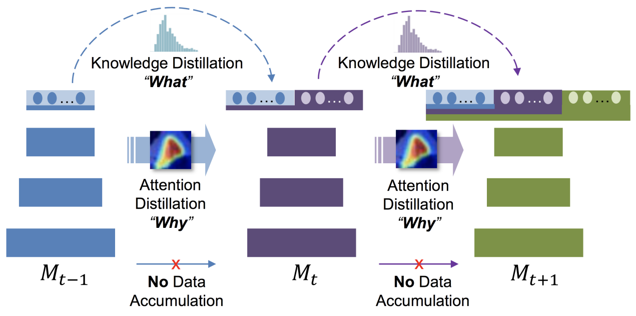

Figure 1: Our problem setup does not store data/model pertaining to information about classes learned in previous incremental steps.

new entities. Inspired by this, we aim to explore incremental learning in object classification by adding a stream of new classes without storing data belonging to classes that the classifier has already seen. While IL solutions that do not require base class data, such as [1, 9] have been proposed, these methods mostly aim at incrementally learning new tasks, which means that at test time the model cannot confuse the incrementally learned tasks with tasks it has already learned, making the problem setup much easier.

We explore the problem of incrementally learning object classes, without storing any data or model associated with the base classes (Figure 1) in the previous steps, while allowing the model to confuse new classes with old ones. In our problem setup, an ideal incremental learner should have the following properties:

i It should help a trained model to learn new classes obtained from a stream of data, while preserving the model's knowledge of base class information.

ii At testing time, it should enable the model to consider all the classes it has learned when the model makes a prediction.

iii The size of the memory footprint should not grow at all, irrespective of the number of classes seen thus far.

An existing work targeting the same problem is LwF-MC, which is one of the baselines in [14]. In the following sections, we use the following terminology (introduced in [19]) at incremental step t :
Teacher model, M_{t-1}, i.e. the model trained with only base classes.
Student model, M_t, i.e. the model which incrementally learns new classes, while emulating the teacher model for maintaining performance on base classes.
Information Preserving Penalty (IPP), i.e. the loss to penalize the divergence between M_{t-1} and M_t. Ideally, this helps M_t to be as proficient in classifying base classes as M_{t-1}.

Initialized using M_{t-1}, M_t is then trained to learn new classes using a classification loss, L_C. However, an IPP is also applied to M_t so as to minimize the divergence between the representations of M_{t-1} and M_t. While L_C helps M_t to learn new classes, IPP prevents M_t from diverging too much from M_{t-1}. Since M_t is already initialized as M_{t-1}, the initial value of IPP is expected to be close to zero. However, as M_t keeps learning new classes with L_C, it starts diverging from M_{t-1}, which leads the IPP to increase. The purpose of the IPP is to prevent the divergence of M_t from M_{t-1}. Once M_t is trained for a fixed number of epochs, it is used as a teacher in the next incremental step, using which a new student model is initialized.

In LwF-MC [14], the IPP is the knowledge distillation loss. The knowledge distillation loss L_D, in this context, was first introduced in [12]. It captures the divergence between the prediction vectors of M_{t-1} and M_t. In an incremental setup, when an image belonging to a new class (I_n) is fed to M_{t-1}, the base classes which have some resemblance in I_n are captured. L_D enforces M_t to capture the same base classes. Thus, L_D essentially makes M_t learn 'what' are the possible base classes in I_n, as shown in Figure 1. Pixels that have significant influence on the models' prediction constitute the attention region of the network. However, L_D does not explicitly take into account the degree of each pixel influencing the models predictions. For example, in Figure 2, in the first row, it is seen that at step n, even though the network focuses on an incorrect region while predicting 'dial_telephone', the numerical value of L_D (0.09) is same as that when the network focuses on the correct region in step n, in the bottom row.

We hypothesize that attention regions encode the models' representation more precisely. Hence, constraining the attention regions of M_t and M_{t-1} using an Attention Distillation Loss (L_D, explained in Sec. 4.1), to minimize the divergence of the representations of M_t from that of M_{t-1} is more meaningful. This is because, instead of finding which base classes are resembled in the new data, attention maps explain 'why' hints of a base class are present (as shown in Figure 1). Using these hints, L_D, in an attempt to make the attention maps of M_{t-1} and M_t equivalent, helps to encode some visual knowledge of base class in M_t. We show the utility of L_{AD} in Figure 2, where although the model correctly predicts the image as 'dial_telephone', the value of L_D in step n increases if the attention regions diverge too much from the region in Step 0.

We propose an approach where an Attention Distillation Loss (L_{AD}) is applied to M_t to prevent its divergence from M_{t-1}, at incremental step t. Precisely, we propose to constrain the L_1 distance between the attention maps generated by M_{t-1} and M_t in order to preserve the knowledge of base classes. The reasoning behind this strategy is described in Sec 4.1. This is applied in addition to the distillation loss L_D and a classification loss for the student model to incrementally learn new classes.

The main contribution of this work is to provide an attention-based approach, termed as 'Learning without Memorizing (LwM)', that helps a model to incrementally learn new classes by restricting the divergence between student and teacher model. LwM does not require any data of the base classes when learning new classes. Different from contemporary approaches that explore the same problem, LwM takes into account the gradient flow information of teacher and student models by generating attention maps using these models. It then constrains this information to be equivalent for teacher and student models, thus preventing the student model to diverge too much from the teacher model. Finally, we show that LwM consis-

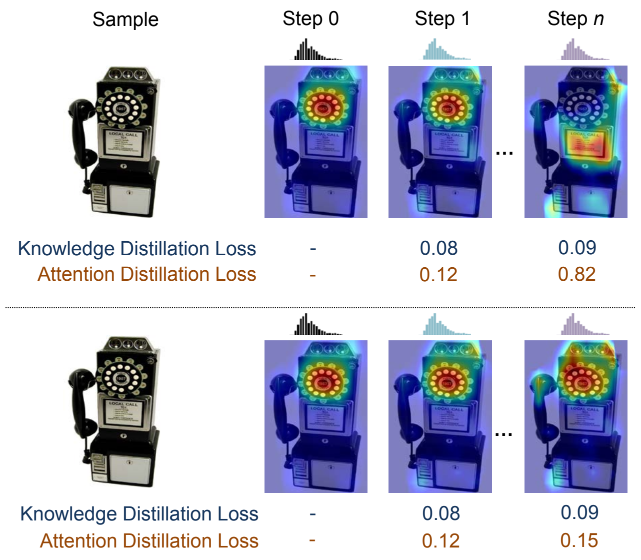
Sample
Step 0
Step 1
Step n
...
Knowledge Distillation Loss
Attention Distillation Loss
-
0.08
0.09
-
0.12
0.82
...
Knowledge Distillation Loss
Attention Distillation Loss
-
0.08
0.09
-
0.12
0.15

Figure 2: (Top) Example of a case where attention regions degrade in later incremental steps.(Bottom) Example of a case where attention regions do not vary across incremental steps. Distillation loss is seen to be unaffected by degrading attention regions, whereas Attention Distillation Loss is sensitive to the attention regions

tently outperforms the state-of-the-art performance in the iILSVRC-small [14] and iCIFAR-100 [14] datasets.

2. Related work

In object classification, Incremental learning (IL) is the process of increasing the breadth of an object classifier, by training it to recognize new classes, while retaining its knowledge of the classes on which it has been trained originally. In the past couple of years, there has been considerable research efforts in this field [9, 12]. Moreover, there exist several subsets of this research problem which impose different constraints in terms of data storage and evaluation. We can divide existing methods based on their constraints:

Task incremental (TI) methods: In this problem, a model trained to perform object classification on a specific dataset is incrementally trained to classify objects in a new dataset. A key characteristic of these experiments is that during evaluation, the final model is tested on different datasets (base and incrementally learned) separately. This is known as multi-headed evaluation [4]. In such an evaluation, the classes belonging to two different tasks have no chance to confuse with one another. One of the earlier works in this category is LwF [12], where a distillation loss is used to preserve information of the base classes. Also, the data from base classes is used during training, while the classifier learns new classes. A prominent work in this area is EWC [9], where at each incremental task the weights of the student model are set to those of their corresponding teacher model, according to their importance of network weights. Aljundi et al. present MAS [1], a technique to train the agents to learn what information should not be forgotten. All experiments in this category use multi-headed evaluation, which is different from the problem setting of this paper where we use single-headed evaluation, defined explicitly in [4]. Single-headed evaluation is another evaluation method wherein the model is evaluated on both base and incrementally learned classes jointly. Multi-headed evaluation is easier than single-headed evaluation, as explained in [4].

Class incremental (CI) methods: In this problem, a model trained to perform object classification on specific classes of a dataset is incrementally trained to classify new unseen classes in the same dataset. Most of the existing work exploring this problem use single-headed evaluation. This makes the CI problem more difficult than the TI problem because the model can confuse the new class with a base class in the CI problem. iCaRL [14] belongs to this category. In iCaRL [14], Rebuffi et al. propose a technique to jointly learn feature representation and classifiers. They also introduce a strategy to select exemplars which is used in combination with the distillation loss to prevent catastrophic forgetting. In addition, a new baseline: LwF-MC is introduced in [14], which is a class incremental version of LwF [12]. LwF-MC uses the distillation loss to preserve the knowledge of base classes along with a classification loss, without storing the data of base classes and is evaluated using single-headed evaluation. Another work aiming to solve the CI problem is [4], which evaluates using both single-headed and multi-headed evaluations and highlights their difference. Chaudhry et al. [4] introduce metrics to quantify forgetting and intransigence, and also propose the Riemannian walk to incrementally learn classes.

A key factor of most incremental learning frameworks is whether or not they allow storing the data of base classes (i.e. classes on which the classifier is originally trained). We can also divide existing methods based on this factor:

Methods which use base class data: Several experiments have been proposed to use a small percentage of the data of base classes while training the classifier to learn new classes. iCaRL [14] uses the exemplars of base classes, while incrementally learning new classes. Similarly, Chaudhry et al. [4] also use a fraction of the data of base classes. Chaudhry et al. [4] also show that this is especially useful for alleviating intransigence, which is a problem faced in single-headed evaluation. However, storing data for base classes increases memory requirement at each incremental step, which is not feasible when the memory budget is limited.

Methods which do not use base class data: Several TI methods described earlier (such as [1, 9]) do not use the information about base classes while training the classifier to learn new classes incrementally. To the best of

Constraints	Use base class data	No base class data
CI methods	iCaRL [14], [4], [18]	LwF-MC [14], **LwM**
TI methods	LwF [12]	IMM [10], EWC [9], MAS [1], [2], [8]

Table 1: Categorization of recent related works in incremental learning. We focus on the class incremental (CI) problems where base class data is unavailable when learning new classes.

our knowledge, LwF-MC [14] is the only CI method which needs no base class data but uses single-headed evaluation.

Table 1 presents a taxonomy of previous works in this field. We propose a technique to solve the CI problem, without using any base class data. We can infer from the discussion above that LwF-MC [14] is the only existing work which uses single-headed evaluation, and hence use it as our baseline. We intend to use attention maps in an incremental setup, instead of only knowledge distillation, to transfer more comprehensive knowledge of base classes from teacher to student model. Although in [19], enforcing equivalence of attention maps of teacher and student models has been explored previously for transferring knowledge from teacher to student models, the same approach cannot be applied to an incremental learning setting. In our incremental problem setup, due to the absence of base class data, we intend to utilize the attention region in the new data which resembles one of the base classes. But these regions are not prominent since the data does not belong to any of the base classes, thus making class-specific attention maps a necessity. Class-specificity is required to mine out base class regions in a more targeted fashion, which is why generic attention maps such as activation-based attention maps in [19] are not applicable as they can not provide a class-specific explanation about relevant patterns corresponding to the target class. We define class-specific interpretation as how a network understands the spatial locations of specific kinds of object. Such locations are determined by computing Grad-CAM [16] attention maps. Also, in LwM, by using class-specific attention map, we can enforce the consistency on class-specific interpretation between teacher and student models. Moreover, our problem setup is different from knowledge distillation because at incremental step t, we freeze M_{t-1} while training M_t, and do not allow M_t to access data from the base classes, and therefore M_{t-1} and M_t are trained using a completely different set of classes. This makes the problem more challenging as the output of M_t on feeding data from unseen classes is the only source of base class data. This is further explained in Sec. 4.1.

We intend to explore the CI problem by proposing to constrain the attention maps of the teacher and student models to be equivalent (in addition to their prediction vectors), to improve the information preserving capability of LwF-MC [14]. In LwF-MC and our proposed method LwM, storing teacher models trained in previous incremental steps is not allowed since it would not be feasible to accumulate models from all the previous steps when the memory budget is limited.

3. Background

Before we discuss LwM, it is important to introduce distillation loss L_D, which is our baseline IPP, as well as how we generate attention maps.

3.1. Distillation loss (L_D)

L_D was first introduced in [12] for incremental learning. It is defined as follows:

$$L_D(\mathbf{y}, \hat{\mathbf{y}}) = -\sum_{i=1}^{N} y_i^{'}.\log(\hat{y_i^{'}}), \tag{1}$$

where $\mathbf{y}$ and $\hat{\mathbf{y}}$ are prediction vectors (composed of probability scores) of M_{t-1} and M_t for base classes at incremental step t, each of length N (assuming that M_{t-1} is trained on N base classes). Also, $y_i^{'} = \sigma(y_i)$ and $\hat{y_i^{'}} = \sigma(\hat{y}_i)$ (where $\sigma(\cdot)$ is sigmoid activation). This definition of L_D is consistent with that defined in LwF-MC [14]. Essentially, L_D enforces the base class prediction of M_t and M_{t-1} to be equivalent, when an image belonging to one of the incrementally added classes is fed to each of them. Moreover, we believe that there exist common visual semantics or patterns in both base and new class data. Therefore, it makes sense to encourage the feature responses of M_t and M_{t-1} to be equivalent, when new class data is given as input. This helps to retain the old class knowledge (in terms of the common visual semantics).

3.2. Generating attention maps

We describe the technique employed to generate attention maps. In our experiments we use the Grad-CAM [16] for this task. In [15], Grad-CAM maps have been shown to encode information to learn new classes, although not in an incremental setup. For using the Grad-CAM, the image is first forwarded to the model, obtaining a raw score for every class. Following this, the gradient of score y^c for a desired class c is computed with respect to each convolutional feature map A_k. For each A_k, global average pooling is performed to obtain the neuron importance α_k of A_k. All the A_k weighted by α_k are passed through a ReLU activation function to obtain a final attention map for class c.

More precisely, let $\alpha_k = \frac{\partial y^c}{\partial A_k}$. Let $\alpha = [\alpha_1, \alpha_2, \dots, \alpha_K]$ and $\mathbf{A} = [A_1, A_2, \dots, A_K]$, where K is the number of convolutional feature maps in the layer using which attention map is to be generated. The attention map Q can be defined as

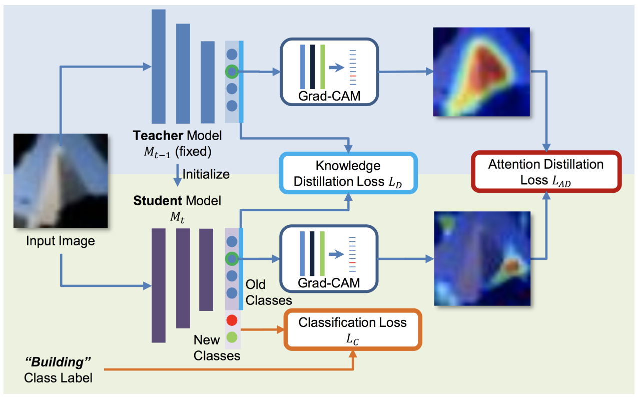

Figure 3: At incremental step t, LwM accepts images belonging to one of the new classes. Three losses (L_C, L_D and L_{AD}) are applied to M_t while M_{t-1} remains frozen. The new classes are depicted in the lower part of the classifier of M_t.

$$Q = ReLU(\alpha^T \mathbf{A}) \tag{2}$$

4. Proposed approach

We introduce an information preserving penalty (L_{AD}) based on attention maps. We combine L_{AD} with distillation loss L_D and a classification loss L_C to construct LwM, an approach which encourages attention maps of teacher and student to be similar. Our LwM framework is shown in Figure 3. The loss function of LwM is defined below:

$$L_{LwM} = L_C + \beta L_D + \gamma L_{AD} \tag{3}$$

Here β, γ are the weights used for L_D, L_{AD} respectively. In comparison to LwM, LwF-MC [14] only uses a classification loss combined with distillation loss and is our baseline.

4.1. Attention distillation loss (L_{AD})

At incremental step t, we define student model M_t, initialized using a teacher model M_{t-1}. We assume M_t is proficient in classifying N base classes. M_t is required to recognize $N + k$ classes, where k is the number of previously unseen classes added incrementally. Hence, the sizes of the prediction vectors of M_{t-1} and M_t are N and $N + k$ respectively. For any given input image i, we denote the vectorized attention maps generated by M_{t-1} and M_t, for class c as $Q_{t-1}^{i,c}$ and $Q_t^{i,c}$, respectively. We generate these maps using Grad-CAM [16], as explained above.

$$Q_{t-1}^{i,c} = vector(\text{Grad-CAM}(i, M_{t-1}, c)) \tag{4}$$

$$Q_t^{i,c} = vector(\text{Grad-CAM}(i, M_t, c)) \tag{5}$$

We assume that the lengths of each vectorized attention map is l. In [19], it has been mentioned that normalizing the attention map by dividing it by the L_2 norm of the map is an important step for student training. Hence we perform this step while computing L_{AD}. During training of M_t, an image belonging to one of the new classes to be learned (denoted as I_n), is given as input to both M_{t-1} and M_t. Let b be the top base class predicted by M_t (i.e. base class having the highest score) for I_n. For this input, L_{AD} is defined as the sum of element wise L_1 difference of the normalized, vectorized attention map:

$$L_{AD} = \sum_{j=1}^{l} \left\| \frac{Q_{t-1,j}^{I_n,b}}{\|Q_{t-1}^{I_n,b}\|_2} - \frac{Q_{t,j}^{I_n,b}}{\|Q_t^{I_n,b}\|_2} \right\|_1 \tag{6}$$

From the explanation above, we know that for training M_t, M_{t-1} is fed with the data from the classes that it has not seen before (I_n). Essentially, the attention regions generated by M_{t-1} for I_n, represent the regions in the image which resemble the base classes. If M_t and M_{t-1} have equivalent knowledge of base classes, they should have a similar response to these regions, and therefore $Q_t^{I_n,b}$ should be similar to $Q_{t-1}^{I_n,b}$. This implies that the attention outputs of M_{t-1} are the only traces of base data, which guides M_t's knowledge of base classes. We use the L_1 distance between $Q_{t-1}^{I_n,b}$ and $Q_t^{I_n,b}$ as a penalty to enforce their similarity. We experimented with both L_1 and L_2 distance in this context. However, as we obtained better results with L_1 distance on held-out data, we chose L_1 over L_2 distance.

According to Eq. 2, attention maps encode gradient of the score of class b, y^b with respect to convolutional feature maps $\mathbf{A}$. This information is not explicitly captured by the distribution of class scores (used by L_D). By encouraging $Q_{t-1}^{I_n,b}$ and $Q_t^{I_n,b}$ to be equivalent, we are restricting the divergence between $\left[\frac{\partial y^b}{\partial \mathbf{A}}\right]_{t-1}$ and $\left[\frac{\partial y^b}{\partial \mathbf{A}}\right]_t$. This ensures the consistency on class-specific interpretation between teacher and student. We know that every feature map in $\mathbf{A}$ encodes a visual feature. While there can be several factors that can cause changes to y^b, L_{AD} forces the changes with respect to a specific visual feature encapsulated in $\mathbf{A}$ to be equivalent for M_t and M_{t-1}. Hence, we hypothesize that combining L_D, which captures the score distribution of the model for base classes $(\mathbf{y}, \hat{\mathbf{y}})$, with a loss that captures the gradient flow information of the model, would result in a more wholesome information preserving penalty. Moreover, the attention maps are a 2D manifestation of the prediction vectors $(\mathbf{y}, \hat{\mathbf{y}})$, which means that they capture more spatial information than these vectors, and hence it is more advantageous to use attention maps than using only prediction vectors.

5. Experiments

We first explain our baseline, which is LwF-MC [14]. Following that, we provide information about the datasets

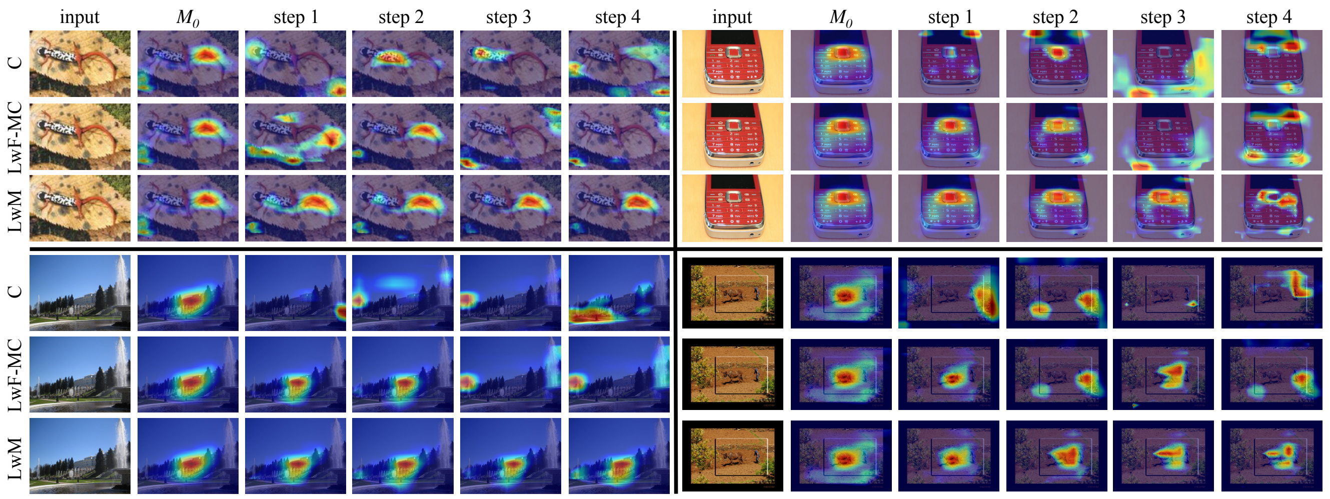

Figure 4: The example attention maps generated by the following experiment IDs (Table 3): C, LwF-MC, and LwM. All the input images belong to the initial base classes. The column M_0 represents the corresponding base-class attention maps generated by the initial teacher model, and the columns step 1∼4 represent the corresponding base-class attention maps generated in four different incremental steps in temporal order. These examples show that the attention maps generated by LwM are closer to those in the column M_0 over time compared with C and LwF-MC, which demonstrates the efficacy of L_{AD} in LwM.

Dataset	iILSVRC -small	iCIFAR -100	CUB200 -2011	Caltech -101
# classes	100	100	100	100
# training images	500	500	80% of data	80% of data
# testing images	100	100	20% of data	20% of data
# classes/batch	10	10, 20, 50	10	10
eval. metric	top-5	top-1	top-1	top-1

Table 2: The statistics of the datasets used in our experiments, in accordance with [14]. Additionally, we also perform experiments on the CUB-200-2011 [17] dataset.

used in our experiments. After that, we describe the iterative protocol to perform classification at every incremental step. We also provide implementation details including architectural information.

5.1. Baseline

As our baseline is LwF-MC [14], we firstly implement its objective function, which is a sum of a classification loss and distillation loss ($L_C + L_D$). In all our experiments, we use a cross entropy loss for L_C to be consistent with [14]. However, it should be highlighted that the official implementation of L_D in LwF-MC by [14] is different from the definition of L_D in [12]. As LwF-MC (but not LwF) is our baseline, we use iCaRL's implementation of LwF-MC in our work. LwF cannot handle CI problems where no base class training data is available (according to Table 1), which is the reason why we choose LwF-MC as the baseline and iCaRL's implementation.

5.2. Datasets

We use two datasets used in LwF-MC [14] for our experiments. Additionally, we also perform experiments on Caltech-101 [5] as well as CUBS-200-2011 [17] datasets. The details for the datasets are provided in Table 2. These datasets are constructed by randomly selecting a batch of classes at every incremental step. In both datasets, the classes belonging to different batches are disjoint. For a fair comparison, the data preparation for all the datsets and evaluation strategy are the same as that for LwF-MC [14].

5.3. Experimental protocol

We now describe the protocol using which we iteratively train M_t, so that it preserves the knowledge of the base classes while incrementally learning new classes.

Initialization: Before the first incremental step ($t = 1$), we train a teacher model M_0 on 10 base classes, using a classification loss for 10 epochs. The classification loss is a cross entropy loss L_C. Following this, for $t = 1$ to $t = k$ we initialize student M_t using M_{t-1} as its teacher, and feed data from a new batch of images that is to be incrementally learned, to both of these models. Here k is the number of incremental steps.

Applying IPP and classification loss to student model: Given the data from new classes as inputs, we generate the output of M_t and M_{t-1} with respect to base class having the highest score. These outputs can either be class-specific attention maps (required for computing L_{AD}) or class-specific scores (required for computing L_D). Using these outputs we compute an IPP which can either be L_{AD} or L_D. In addition, we apply a classification loss to M_t

Experiment ID\loss	L_C	L_D	L_{AD}
Finetuning	✓	✗	✗
LwF-MC [14]	✓	✓	✗
LwM	✓	✓	✓

Table 3: Experiment configurations used in this work, identified by their respective experiment IDs.

based on its outputs with respect to the new classes which are to be learned incrementally. We jointly apply classification loss and IPP to M_t and train it for 10 epochs. Once M_t is trained, we use it as a teacher model in the next incremental step, and follow the aforementioned steps iteratively, until all the k incremental steps are completed.

5.4. Implementation details

We use the ResNet-18 [7] architecture for training student and teacher models on the iILSVRC-small, Caltech-101 and CUBS-200-2011 datasets, and the ResNet-34 [7] for training models on the iCIFAR-100 dataset. This is consistent with the networks and datasets used in [14]. We used a learning rate of 0.01. The feature maps of the final convolutional layer are used to generate attention maps using Grad-CAM, as these maps are highly interpretable. [16]. The combinations of classification loss and IPP, along with their experiment IDs are provided in Table 3. The experiment configurations will be referred to as their respective experiment IDs from now on.

6. Results

Before discussing the quantitative results and advantages of our proposed penalties, we show some qualitative results to demonstrate the advantage of using L_{AD}. We show that we can retain attention regions of base classes for a longer time when more classes are incrementally added to the classifier by using LwM as compared to LwF-MC [14]. Before the first incremental step $t = 1$, we have M_0 trained on 10 base classes. Now, following the protocol in Sec. 5.3, we incrementally add 10 classes at each incremental step. At every incremental step t, we train M_t with 3 configurations: C, LwF-MC [14], and LwM. We use M_t to generate the attention maps for the data from base classes (using which M_0 was trained), which it has not seen, and show the results in Figure 4. Additionally, we also generate corresponding attention maps using M_0 (i.e. the first teacher model), which can be considered 'ideal' (as target maps) as M_0 was given full access to base class data. For the M_ts trained with C, it is seen that attention regions for base classes are quickly forgotten after every incremental step. This can be attributed to catastrophic forgetting [9, 10]. M_t trained with LwF-MC [14] have slightly better attention preserving ability but as the number of incremental steps increases, the attention regions diverge from the 'ideal' attention regions.

# Classes	FT	LwM (ours)	FT	LwM (ours)
Dataset	Caltech-101		CUBS-200-2011	
10 (base)	97.78	97.78	99.17	99.17
20	59.55	**75.34**	57.92	**78.75**
30	52.65	**71.78**	41.11	**70.83**
40	44.51	**67.49**	35.42	**58.54**
50	35.52	**59.79**	32.33	**53.67**
60	31.18	**56.62**	29.03	**47.92**
70	32.99	**54.62**	22.14	**43.79**
80	27.45	**48.71**	22.27	**43.83**
90	28.55	**46.21**	20.52	**39.85**
100	28.26	**48.42**	17.4	**34.52**

Table 4: Results obtained on Caltech-101 [5] and CUBS-200-2011 [17]. Here FT refers to finetuning. The first step refers to the training of first teacher model using 10 classes.

# Classes / Config	$L_C + L_{AD}$	LwM (ours)
20	84.95	**99.55**
30	55.82	**99.18**
40	43.46	**98.72**
50	36.36	**98.10**
60	26.78	**97.22**

Table 5: Top-5 accuracy comparison of L_C + L_{AD} and LwM. The LwM accuracies are in accordance to that of Figure 5. Not designed to be used alone, L_{AD} is used to ensure the consistency on class-specific interpretation between teacher and student, by enforcing the gradients of class-specific score w.r.t. feature maps to be equivalent.

Interestingly, the attention maps generated by M_t trained with LwM configuration retain the attention regions for base classes for all incremental steps shown in Figure 4, and are most similar to the target attention maps. These examples support that LwM delays forgetting of base class knowledge.

We now present the quantitative results of the following configurations: C , LwF-MC [14] and LwM. To show the efficacy of LwM across, we evaluate these configurations on multiple datasets. The results on the iILSVRC-small and iCIFAR-100 datasets are presented in Figure 5. For the iILSVRC-small dataset, the performance of LwM is better than that of the baseline LwF-MC [14]. LwM outperforms the baseline by a margin of more than 30% when the number of classes is 40 or more. Especially for 100 classes, LwM achieves an improvement of more than 50% over the baseline LwF-MC [14]. In addition, LwM outperforms iCaRL [14], at every incremental step, even though iCaRL has the unfair advantage of storing the exemplars of base classes while training the student model for the iILSVRC-small dataset.

To be consistent with the LwF-MC experiments in [14], we perform experiments by constructing the iCIFAR-100

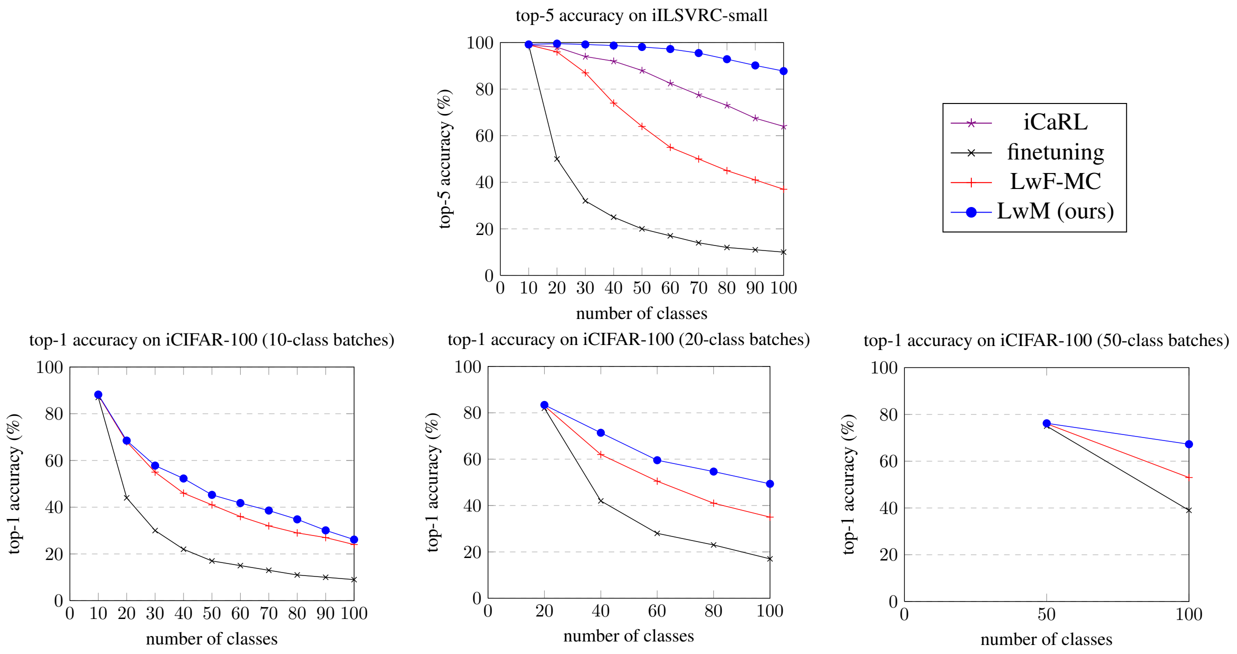

Figure 5: Performance comparison between our method, LwM, and the baselines. LwM outperforms LwF-MC [14] and "using only classification loss with finetuning" on the iILSVRC-small and iCIFAR-100 datasets [14]. LwM even outperforms iCaRL [14] on the iILSVRC-small dataset given that iCaRL has the unfair advantage of accessing the base-class data.

datasets by using batches of 10, 20, and 50 classes at each incremental step. The results are provided in Figure 5. It can be seen that LwM outperforms LwF-MC for all three sizes of incremental batches in iCIFAR-100 dataset. Hence, we conclude that LwM consistently outperforms LwF-MC [14] in iILSVRC-small and iCIFAR-100 datasets. Additionally, we also perform these experiments using the Caltech-101 and CUBS-200-2011 dataset [5] by adding a batch of 10 classes at every incremental step and compare it with finetuning. The results for these two datasets are shown in Table 4.In Table 5, we also provide the results obtained using only a combination of L_C and L_{AD}, on a few incremental steps in iILSVRC-small dataset.

The advantage of incrementally adding every loss on top of L_C is demonstrated in Figure 5, where we show that the performance with only C is poor due to the catastrophic forgetting [9, 10]. We achieve some improvement when L_D is added as an IPP in LwF-MC. The performance further improves with the addition of L_{AD} in LwM configuration.

7. Conclusion and future work

We explored the IL problem for the task of object classification, and proposed a technique: LwM by combining L_D with L_{AD}, for utilizing attention maps to transfer the knowledge of base classes from the teacher to student model, without requiring any data of base classes during training. This technique outperforms the baseline in all the scenarios that we investigate. Regarding future applications, LwM can be used in many real world scenarios. While we explore IL problem for classification in this work, we believe that the proposed approach can also be extended for segmentation. Incremental segmentation is a challenging problem due to the absence of abundant ground truth maps. The importance of incremental segmentation has already been underscored in [3]. As visual attention is also meaningful for segmentation (as shown in [11]), we intend to extend LwM to incremental segmentation in the near future.

Acknowledgment

Mostly done when Prithviraj was an intern at Siemens, this work was partially supported by the Intelligence Advanced Research Projects Activity (IARPA) via Department of Interior/Interior Business Center (DOI/IBC) contract number D17PC00345. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes not withstanding any copyright annotation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied of IARPA, DOI/IBC or the U.S. Government.

References

[1] R. Aljundi, F. Babiloni, M. Elhoseiny, M. Rohrbach, and T. Tuytelaars. Memory aware synapses: Learning what (not) to forget. In *The European Conference on Computer Vision (ECCV)*, 2018.

[2] R. Aljundi, M. Rohrbach, and T. Tuytelaars. Selfless sequential learning. *arXiv preprint arXiv:1806.05421*, 2018.

[3] C. Baweja, B. Glocker, and K. Kamnitsas. Towards continual learning in medical imaging. In *Medical Imaging meets NIPS Workshop*, 2018.

[4] A. Chaudhry, P. K. Dokania, T. Ajanthan, and P. H. S. Torr. Riemannian walk for incremental learning: Understanding forgetting and intransigence. In *The European Conference on Computer Vision (ECCV)*, September 2018.

[5] L. Fei-Fei, R. Fergus, and P. Perona. One-shot learning of object categories. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28:594–611, 2006.

[6] R. Girshick. Fast R-CNN. In *Proceedings of the IEEE international conference on computer vision*, pages 1440–1448, 2015.

[7] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[8] H. Jung, J. Ju, M. Jung, and J. Kim. Less-forgetting learning in deep neural networks. In *AAAI*, 2018.

[9] J. Kirkpatrick, R. Pascanu, N. Rabinowitz, J. Veness, G. Desjardins, A. A. Rusu, K. Milan, J. Quan, T. Ramalho, A. Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, page 201611835, 2017.

[10] S.-W. Lee, J.-H. Kim, J. Jun, J.-W. Ha, and B.-T. Zhang. Overcoming catastrophic forgetting by incremental moment matching. In *Advances in Neural Information Processing Systems*, pages 4652–4662, 2017.

[11] K. Li, Z. Wu, K.-C. Peng, J. Ernst, and Y. Fu. Tell me where to look: Guided attention inference network. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[12] Z. Li and D. Hoiem. Learning without forgetting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017.

[13] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431–3440, 2015.

[14] S.-A. Rebuffi, A. Kolesnikov, G. Sperl, and C. H. Lampert. iCaRL: Incremental classifier and representation learning. In *Proc. CVPR*, 2017.

[15] R. R. Selvaraju, P. Chattopadhyay, M. Elhoseiny, T. Sharma, D. Batra, D. Parikh, and S. Lee. Choose your neuron: Incorporating domain knowledge through neuron-importance. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 526–541, 2018.

[16] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, D. Batra, et al. Grad-CAM: Visual explanations from deep networks via gradient-based localization. In *ICCV*, pages 618–626, 2017.

[17] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. The Caltech-UCSD Birds-200-2011 Dataset. Technical Report CNS-TR-2011-001, California Institute of Technology, 2011.

[18] Y. Wu, Y. Chen, L. Wang, Y. Ye, Z. Liu, Y. Guo, Z. Zhang, and Y. Fu. Incremental classifier learning with generative adversarial networks. *arXiv preprint arXiv:1802.00853*, 2018.

[19] S. Zagoruyko and N. Komodakis. Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer. In *ICLR*, 2017.