Sharpness and Gradient Aware Minimization for Memory-based Continual Learning

Tran Tung Lam
School of Information and Communication Technology,
Hanoi University of Science and Technology
Hanoi, Vietnam
tunglamlqddb@gmail.com

Nguyen Van Viet
School of Information and Communication Technology,
Hanoi University of Science and Technology
Hanoi, Vietnam
viettmab123@gmail.com

Nguyen Hoang Phi
School of Information and Communication Technology,
Hanoi University of Science and Technology
Hanoi, Vietnam
hoangphi4589@gmail.com

Khoat Than
School of Information and Communication Technology,
Hanoi University of Science and Technology
Hanoi, Vietnam
khoattq@soict.hust.edu.vn

ABSTRACT

Memory-based Continual Learning methods (CL) preserve performance on old data by storing a small buffer of seen samples to re-learn with current data. Despite their impressive results, these methods may still obtain sub-optimal solutions as a result of overfitting training data, especially on the limited buffer. This can be attributed to their employment of empirical risk minimization over training data. To overcome this problem, we leverage Sharpness Aware Minimization (SAM), a recently proposed training technique, to improve models' generalization, and thus CL performance. In particular, SAM seeks for flat minima whose neighbors' loss values are also low by simultaneously guiding a model towards SAM gradient direction corresponding to low-loss regions and flat regions. However, we conjecture that directly applying SAM to replay-based CL methods whose loss function contains multiple objectives may cause gradient conflict among them. We then propose to manipulate each objective's SAM gradient such that their potential conflict is minimized by adopting one gradient aggregation strategy from Multi-task Learning. Finally, through extensive experiments, we empirically verify our hypothesis and show consistent improvements in our method over strong memory-replay baselines.

CCS CONCEPTS

• **Computing methodologies → Object recognition**.

KEYWORDS

Continual Learning, Flatness, Gradient Alignment

ACM Reference Format:
Tran Tung Lam, Nguyen Van Viet, Nguyen Hoang Phi, and Khoat Than. 2023. Sharpness and Gradient Aware Minimization for Memory-based Continual Learning. In *The 12th International Symposium on Information and Communication Technology (SOICT 2023), December 07–08, 2023, Ho Chi Minh, Vietnam.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3628797.3629000

SOICT 2023, December 07–08, 2023, Ho Chi Minh, Vietnam

ACM ISBN 979-8-4007-0891-6/23/12…$15.00
https://doi.org/10.1145/3628797.3629000

1 INTRODUCTION

The ability to acquire new knowledge without forgetting previous information, is a crucial challenge for deep learning models. These models often excel with static, identical independent data, but struggle when faced with sequential and dynamically changing tasks, such as self-driving cars in varying environments. In this case, the model will inevitably witness an abrupt decline in its performance on the first task, a phenomenon called catastrophic forgetting [10]. To tackle this issue, Continual Learning (CL) has gained significant attention in the community.

Existing CL methods are often divided into three main groups: architecture-based [23, 25], regularization-based [13, 15, 29], and memory-based approaches [4–6, 22]. Methods in the first group often either allocate a new part or spare a subset of the network for each new task, resulting in the inevitable network expansion or network insufficiency as more tasks arrive. However, the other two groups can deal with CL using a fixed-size network. Notably, memory-based, or memory replay, methods often achieve state-of-the-art results by storing a limited size of examples representing old tasks to replay with data from the current task. These examples can be in the original space [5], in hidden space [2], or in gradient space [7, 24]. The replay strategies can be designed in many different sophisticated ways [1, 14, 19]. Nevertheless, the majority of this group follows one simple principle when designing the objective function: combining the current task's loss with a specific loss computed on the buffer, e.g. cross-entropy or $l2$ regularization on hidden features, to retain old knowledge while learning new one. Usually, these losses are considered in the empirical form, i.e. the average over training data points, so the methods tend to share one common drawback: overfitting on training data [8], notably to the limited size buffer.

In general, the overfitting problem of neural networks is a result of a high-dimensional non-convex loss function whose landscape is complex and contains many local optima [12, 21]. Therefore, to achieve a more generalized model, seeking flat local minima has been one of the most effective approaches [8, 17, 30]. This idea of

a flat-seeking optimizer has been applied to CL [3, 7, 26]. Broadly speaking, they observe that a model converging to a flat local minimum tends to find common optimal regions for different tasks more easily than one that converges to sharp minima, thus it obtains better CL performance. Similarly, in this work, we investigate the impact of a recently introduced sharpness aware minimization, SAM, on memory replay methods, the ingredient of a competitive CL solver. Instead of following standard training that optimizes an empirical loss, SAM solves the worst-case one in the neighborhood of current model parameters. It is worth noting that Deng et al. [7] also studied SAM in their work. However, they applied SAM to their main method which is a gradient-based one, and their results on an exemplar replay baseline are limited. Moreover, their reported results were under task-incremental learning setting (TIL), i.e. we know which task a test sample belongs to (known task identity).

Contributions:

In line with FS-GPM [7], we *first* propose to incorporate SAM into CL (**SAM-CL**) to boost model performance. However, we differently apply SAM to a simple yet general memory-based baseline, and additionally conduct experiments under a class-incremental learning setting (CIL), a more practical and difficult scenario when no task identity is given during inference [18]. Thus, our work not only can be easily extended to other complex memory-based methods whose core objective function is similar to our chosen baseline but also can perform under a popular CL setting, CIL. Nevertheless, combined with FS-GPM, ours can provide a comprehensive analysis of the benefit of SAM on CL. *Furthermore*, after analyzing the SAM gradients of each term in the objective function, we conjecture and empirically verify that the may exist conflict between these gradients, which can hinder the capability of the model to converge to flat minima. This drives us to our final proposed method: Sharpness Gradient Aware Minimization in CL, **SGAM-CL**, that first finds the SAM gradient of each loss term, then combines them using a gradient aggregation strategy to reduce potential conflict. *Finally*, through extensive experiments on four common benchmarks, we show that SGAM-CL brings a significant improvement for the baseline, and verify the further enhancement of SGAM-CL over SAM-CL.

2 RELATED WORK

2.1 Three main Continual learning approaches

Expansion-based methods. assign separate subsets of parameters to each task to avoid interference. For example, Progressive Neural Networks [23] learn lateral connections to prior tasks to leverage existing features while allocating new parameters to each task. Alternatively, with a fixed-size network, Hard attention to the tasks [25] learns a task-specific mask over neurons and restricts the update of gradients connecting two important neurons. However, they require expanding the model complexity and knowing task boundaries during training, thus not applicable under CIL.

Regularization-based methods. add regularization terms to the loss function to restrict changes in weights that are important for old tasks. Elastic Weight Consolidation [13] is a seminal method that penalizes weight updates based on their Fisher information to preserve performance on old tasks. Another line of works, inspired by Knowledge Distillation, forces outputs of the current network to be similar to those of the previous one, with Learning without Forgetting [15] as a notable example. This method however performs poorly under CIL setting when saving no past data.

Memory-based methods. store data from previous tasks and replay them during new task learning. Exemplar-replay [6] uses reservoir sampling to select incoming data for the buffer. iCaRL [22] saves data closest to the feature mean of each class. Gradient Episodic Memory (GEM) [16] and Averaged GEM [5] store a subset of data and constrain gradient updates to not increase loss on replayed data. When access to raw data is not allowed or more old data is required, generative replay approaches like Deep Generative Replay [27] use generative models to produce synthetic samples from previous tasks. Recently, Dark Experience Replay [2], whose objective is a mixture of distillation, regularization, and replay, obtains a strong yet simple baseline. Based on the fact that the majority of CIL competitive methods share this same loss, a further boost to this simple baseline would benefit the community.

2.2 Flat minima in Continual Learning

Flat-seeking minimizers have been extensively studied theoretically and empirically due to their ability to improve model generalization in single-task learning [8, 12, 17]. Inspired by this, several works attempt to investigate the merits of flat minima in CL [2, 3, 7, 20]. For instance, in [20], forgetting of the first task is approximated using Taylor expansion suggests from which a wider minimum is proven to forget less. It then proposes to control batch size, learning rate, and dropout during training as these elements have been proven to promote wider local minima [12]. Additionally, CPR [3] enforces high entropy of softmax outputs to reduce forgetting of regularization-based methods under the TIL scenario. Later, in few-shot CIL, F2M [26] introduces flatness to the first task training of base classes so that good solutions for novel tasks can be found in such flat regions. Recently, FS-GPM [7] studies the dilemma between learning new tasks and memorizing old ones from the weight loss landscape viewpoint. They similarly find that solutions with low loss values and flat landscapes can lead to better CL performance, then apply SAM to Gradient Projection Memory, a method requiring task-id to achieve good results.

Compared with our method, CPR [3] adds an entropy penalty to the objective of existing regularization-based methods which may make the model sensitive to hyper-parameters as a higher weight for the entropy term can hinder the learning of the current task, while a lower value may not guarantee a flat minimum. Although F2M [26] also promotes flatness by penalizing the loss value of current parameters and those of close neighbors, it achieves this by randomly adding noises to the current parameters and jointly minimizing losses of obtained models. This approximation in practice is costly or may not find flat minima if the number of random sampling is small to maintain feasible training time. Lastly, as mentioned earlier, the investigation in FS-GPM [7] does not cover the CIL scenario, and mainly shows the benefit of SAM to gradient-based methods. In this paper, we aim to present the superiority of SAM over previous flat minimizers when applied to a strong CL approach, replay-based, and under a more realistic scenario, CIL.

3 PRELIMINARIES

3.1 Continual Learning

In this section, we are going to introduce the concept of incremental learning, or continual learning problem (CL). Specifically, we denote T sequential incremental tasks as $T_1, T_2, \ldots, T_T$, where each task includes a set of disjoint classes. For the t^{th} task, we denote $D_t = \{(x_{t,i}, y_{t,i})\}_{i=1}^{N_t}$ to represent the dataset corresponding to this task, where N_t is the number of samples in the dataset. We focus on the classification problem to be in line with the setting of many existing CL methods.

In addition, we have a model parameterized by $\boldsymbol{w} = (\boldsymbol{\theta}, \{\boldsymbol{\phi}_t\}_{t=1}^{T})$, where $\boldsymbol{\theta}$ represents the parameter set of the model's feature extractor module f, and $\boldsymbol{\phi}_t$ is the parameters of the classifier module g for the classes belonging to task t. At the beginning of the t^{th} task, given the model $\{\boldsymbol{\theta}^*_{t-1}, \boldsymbol{\phi}^*_{1:t-1}\}$ previously trained on past tasks, and only data D_t, we need to update the model to obtain a new one $\{\boldsymbol{\theta}^*_t, \boldsymbol{\phi}^*_{1:t}\}$. such that it can satisfy two inherent properties:

1) **Stability**: The new model should retain the good performance on the previous $t-1$ tasks.

2) **Plasticity**: The new model should have the ability to quickly adapt to the t^{th} task.

The latter property can be achieved by optimizing the model with the commonly used classification loss function on D_t:

$$\mathcal{L}_{D_t}(\boldsymbol{w}) := \mathbb{E}_{(x,y)\sim D_t}\ell(g(f(x,\boldsymbol{\theta}),\boldsymbol{\phi}_t),y) \tag{1}$$

where $\ell(.,.)$ is the cross-entropy loss function (CE). However, the former property is challenging as we have no access to previous data, and if the model is trained with Eq. 1 only, it will face catastrophic forgetting, i.e. its classification performance on past classes degrading significantly. To overcome this, we follow the memory-based approach to store a small subset of past data in a buffer $B = \{M_i\}_{i=1}^{t-1}$. In this scenario, many sophisticated methods to retain the previously acquired knowledge encapsulated in $\{\boldsymbol{\theta}^*_{t-1}, \boldsymbol{\phi}^*_{1:t-1}\}$ and B have been developed [14]. Despite this, their objective functions mostly follow the same core design: combining the CE loss on the current task's data with CE loss on buffer data and a knowledge distillation (KD) loss between the current model and the model trained after task $t-1$, represented as follows:

$$\mathcal{L}_{total} := \mathcal{L}_{D_t}(\boldsymbol{w}) + \mathcal{L}_B(\boldsymbol{w}) + KD_B(\boldsymbol{w}, \boldsymbol{w}^*_{t-1}), \tag{2}$$

where $KD_B(\boldsymbol{\theta}, \boldsymbol{\theta}^*_{t-1})$ can be the L_2 regularization on the feature space, i.e. $\|f(x,\boldsymbol{\theta}) - f(x,\boldsymbol{\theta}^*_{t-1})\|_2$, or the KL divergence on the output space, i.e. $\mathcal{D}_{KL}(\text{softmax}(o)||\text{softmax}(o^*_{t-1})), o = g(f(x,\boldsymbol{\theta}),\boldsymbol{\phi}_{1:t-1})$, In addition to using CE loss only (the second term), being inspired by knowledge distillation, the third loss term is introduced to force the current model to mimic the knowledge of its teacher, i.e. the previous model [2, 11], thus further reducing forgetting. It should be noted that the design choices for this term can be varied. For instance, DER [2] uses the L_2 regularization on the logit space and additionally saves samples' logits along the training trajectory to the buffer. These logits are then treated as targets for the current model's logits to match, instead of using logits generated from the previous model as in conventional KD.

However, Eq.6 simply minimizes the empirical losses, i.e. the average loss over data points, on D_t as well as B, which can easily cause the model to overfit these data sets, especially on B because of its limited size $|B| \ll |D_{1-t}|$. This is detrimental to remembering past tasks as the model cannot generalize well to other old samples rather than ones in B. Thus a solution to mitigate this overfitting problem is required.

3.2 Sharpness Aware Minimization

Traditional training techniques, which concentrate on reducing the empirical loss, are susceptible to overfitting issues. This occurs when the validation error plateaus even though the training loss continues to drop, leading to a limitation in the model's ability to generalize. To address this, Foret et al. [9] suggested a method that involves minimizing the worst-case loss within a neighborhood of the current model parameters. Specifically, assume our model has parameters set $\boldsymbol{w}$, then the worst-case loss is:

$$\min_{\boldsymbol{w}} \max_{\|\boldsymbol{\epsilon}\|_2\leq\rho} \mathcal{L}_D(\boldsymbol{w}+\boldsymbol{\epsilon}), \tag{3}$$

with ρ is the radius perturbation of the current parameter $\boldsymbol{w}$.

The optimal solution of the inner optimization in Eq.(3) can be approximated as:

$$\boldsymbol{\epsilon}^* := \arg\max_{\|\boldsymbol{\epsilon}\|_2\leq\rho} \mathcal{L}_D(\boldsymbol{w}+\boldsymbol{\epsilon}) \approx \rho.\frac{\nabla_{\boldsymbol{w}}\mathcal{L}_D(\boldsymbol{w})}{\|\nabla_{\boldsymbol{w}}\mathcal{L}_D(\boldsymbol{w})\|_2}.$$

This leads to the gradient update of the model:

$$\boldsymbol{g}^{SAM} := \nabla_{\boldsymbol{w}} \max_{\|\boldsymbol{\epsilon}\|_2\leq\rho} \mathcal{L}_D(\boldsymbol{w}+\boldsymbol{\epsilon}) \approx \nabla_{\boldsymbol{w}}\mathcal{L}_D(\boldsymbol{w})|_{\boldsymbol{w}+\boldsymbol{\epsilon}^*}. \tag{4}$$

3.3 Projecting conflicting gradients - PCGrad

As mentioned earlier, one of our contributions is to resolve potential conflicts between SAM gradients of each loss term in the objective function. To do this, we propose to apply a simple gradient aggregation technique in multi-task learning MTL, PCGrad [28]. To make the paper self-contained, we will briefly introduce the issue of gradient conflict in MTL and how PCGrad mitigates this conflict.

In multi-task learning, we aim to concurrently learn n tasks with the corresponding objective functions: $\mathcal{L}_1(w), \cdots, \mathcal{L}_n(w)$, which is normally done by minimizing the loss: $\min_{\boldsymbol{w}} \sum_{i=1}^{n} \mathcal{L}_i(w)$. However, in practice, this minimization does not guarantee to decrease losses of all tasks due to the possible conflict between tasks. Particularly, some tasks may dominate others, leading to the improvement of some tasks while the rest suffer [28]. Formally, two tasks conflict when the cosine similarity between their gradients is less than 0: $\langle \boldsymbol{g}_i.\boldsymbol{g}_j < 0 \rangle$, where $\langle .,. \rangle$ is the dot product. This means task gradients might cancel out each other or point to a direction where the performance of one task will be hurt.

PCGrad resolves the disagreement between tasks by projecting gradients that conflict with each other, i.e. $\langle \boldsymbol{g}_i, \boldsymbol{g}_j \rangle < 0$, to the orthogonal direction of each other. Specifically, $\boldsymbol{g}_i$ is replaced by its projection on the normal plane of $\boldsymbol{g}_j$:

$$\bar{\boldsymbol{g}}_i = \boldsymbol{g}_i - \frac{\boldsymbol{g}_i \cdot \boldsymbol{g}_j}{||\boldsymbol{g}_j||^2}\boldsymbol{g}_j. \tag{5}$$

Then the aggregated gradient is computed based on these deconflicted vectors $\mathbf{g} = \sum_i^n \bar{\boldsymbol{g}}_i$.

4 PROPOSED METHOD

In this section, we will first present the direct application of SAM on a simple memory-based CL method to enable the flatness of the learned model. Next, we will show our conjecture about the potential conflict between SAM gradients of the loss terms, which might hinder the model from finding a region where all losses converge to a flat local minimum. Finally, our proposed solution, SAM-CL, which aims to reduce this conflict, will be presented.

4.1 Sharpness Aware Minimization for Memory replay methods

As aforementioned, our focus is the benefit of SAM on a memory replay method whose loss function consists of the core terms on which many complex and sophisticated methods are based. In particular, our choice for the objective is:

$$\mathcal{L}_{total} \coloneqq \mathcal{L}_{D_t}(\boldsymbol{w}) + \alpha \mathcal{L}_B(\boldsymbol{w}) + \beta \|f(x, \boldsymbol{\theta}) - f(x, \boldsymbol{\theta}^*_{t-1})\|_2, \quad (6)$$

where α, β are two weighting terms. To apply SAM, the most direct way is to treat $\mathcal{L}_{total}$ as the loss function $\mathcal{L}_D$ in Eq. (4) to find the SAM gradient, then update the model:

$$\boldsymbol{\epsilon}^* = \rho . \frac{\nabla_{\boldsymbol{w}} \mathcal{L}_{total}(\boldsymbol{w})}{\|\nabla_{\boldsymbol{w}} \mathcal{L}_{total}(\boldsymbol{w})\|_2},$$
$$\boldsymbol{g}^{SAM} = \nabla_{\boldsymbol{w}} \mathcal{L}_{total}(\boldsymbol{w})|_{\boldsymbol{w}+\boldsymbol{\epsilon}^*}, \quad \boldsymbol{w} = \boldsymbol{w} - \eta \boldsymbol{g}^{SAM}, \quad (7)$$

where $\eta > 0$ is the learning rate. As a result, the objective value $\mathcal{L}_{total}$ of the obtained model will not change significantly against small local changes in the parameter space, i.e. flat minima, thus CL performance can be enhanced, which has been proven theoretically and empirically in previous works [3, 20]. However, we argue that this enhancement, brought by the direct approach, can still be further improved if we take into account the directions of gradients of each loss term (currently only their magnitudes are considered through α and β). This is because these gradients define where the model will update to reach flat minima. Supposing the gradient of one term dominates or cancels out those of the others, its flat minima may be favored over other terms'. Thus, the obtained minima may not be evenly flat for each loss term, or for current and buffer data in our case. Our formal conjecture for this hidden risk of the direct approach will be presented below.

4.2 Potential conflict of worst-case gradients

To begin with, let us re-write $\boldsymbol{g}^{SAM}$ in a more intuitive way using first-order Taylor expansion:

$$\begin{aligned}
\boldsymbol{g}^{SAM} &= \nabla_{\boldsymbol{w}} \mathcal{L}_{total}(\boldsymbol{w} + \boldsymbol{\epsilon}^*) \\
&\approx \nabla_{\boldsymbol{w}} \left[\mathcal{L}_{total}(\boldsymbol{w}) + \langle \boldsymbol{\epsilon}^*, \nabla_{\boldsymbol{w}} \mathcal{L}_{total}(\boldsymbol{w}) \rangle \right] \\
&= \nabla_{\boldsymbol{w}} \left[\mathcal{L}_{total}(\boldsymbol{w}) + \rho \langle \frac{\nabla_{\boldsymbol{w}} \mathcal{L}_{total}(\boldsymbol{w})}{\|\nabla_{\boldsymbol{w}} \mathcal{L}_{total}(\boldsymbol{w})\|_2}, \nabla_{\boldsymbol{w}} \mathcal{L}_{total}(\boldsymbol{w}) \rangle \right] \\
&= \nabla_{\boldsymbol{w}} \left[\mathcal{L}_{total}(\boldsymbol{w}) + \rho \|\nabla_{\boldsymbol{w}} \mathcal{L}_{total}(\boldsymbol{w})\|_2 \right] \quad (8)
\end{aligned}$$

This equation shows that the SAM gradient will follow two directions to optimize the model: (i) direction that minimizes the original loss $\mathcal{L}_{\boldsymbol{w}}$, and (ii) direction that minimizes norm of the gradient. The first direction accounts for finding regions with low loss value, and the second one finds flat regions, thus together leading the model to the flat minima.

Considering our case where the total loss contains two loss terms: one aims to find local flat minima w.r.t current data, and the other seeks for those minima w.r.t buffer data (the second and third terms are combined as they are all computed on buffer). Combined with the above analysis, it can be seen that each loss term has two directions, low-loss and flat regions, to follow, making the SAM gradient have to aggregate from multiple directions which may interfere with each other. Therefore, we hypothesize, and will empirically validate in the experiment section, that there may exist gradient conflict between the two terms so that the process in Eq.(7) might not guarantee finding the common flat minima for both current and buffer data. Such a case can happen when there exists at least one gradient that interferes or cancels out the others. For example, if the current task is about classifying dogs and cars, then the model may take the four-legged feature to recognize a dog. However, that feature may fail to classify two past classes: cats and deer as they are both four-legged animals. Thus if the model follows directions that benefit current data, it may harm the performance of previous tasks.

4.3 Sharpness and Gradient Aware for memory replay methods

To circumvent the aforementioned pitfall, we propose to separately compute SAM gradients of each loss term, then aggregate them such that their conflict is minimized, and finally update the model with the aggregated gradient. By doing this, the model is encouraged to learn features that are useful for all tasks.

The steps are described as follows: For each loss term $\mathcal{L}_i$:

$$\boldsymbol{\epsilon}^*_i = \rho . \frac{\nabla_{\boldsymbol{w}} \mathcal{L}_i(\boldsymbol{w})}{\|\nabla_{\boldsymbol{w}} \mathcal{L}_i(\boldsymbol{w})\|_2}, \; \boldsymbol{g}^{SAM}_i = \nabla_{\boldsymbol{w}} \mathcal{L}_i(\boldsymbol{w} + \boldsymbol{\epsilon})|_{\boldsymbol{w}+\boldsymbol{\epsilon}^*}. \quad (9)$$

Then apply PCGrad on these SAM gradients to get the final gradient for the model update:

$$\bar{\boldsymbol{g}} = PCGrad(\boldsymbol{g}^{SAM}_1, \boldsymbol{g}^{SAM}_2, \boldsymbol{g}^{SAM}_3), \quad \boldsymbol{w} = \boldsymbol{w} - \eta \bar{\boldsymbol{g}}. \quad (10)$$

In our method, we choose PCGrad as a representative gradient aggregation. We note that our method is agnostic to the application of other aggregation strategies. We leave further exploration on the impact of different aggregation strategies on CL for future work.

Our method, SAM-CL, is summarized in Algorithm 1.

Algorithm 1 SGAM-CL

Input: Model parameter $\boldsymbol{w} = \{\boldsymbol{\theta}, \boldsymbol{\phi}_{1:t}\}$, perturbation radius ρ, step size η, current dataset D_t, buffer $B = \{M_i\}_{i=1}^{t-1}$. **Output:**Updated parameter $\boldsymbol{\theta}^*, \boldsymbol{\phi}^*$

1: **for** $bx \in D_t$ **do**
2: - Sample obx samples from B
3: - Objective function:
$\mathcal{L}_{total} = \mathcal{L}_{bx}(\boldsymbol{\theta}, \boldsymbol{\phi}_t) + \mathcal{L}_{obx}(\boldsymbol{\theta}, \boldsymbol{\phi}_{1:t}) + \mathrm{KD}_{obx}(\boldsymbol{\theta}, \boldsymbol{\theta}^*_{t-1})$.
4: - For each term in the loss, calculate the SAM gradient: $\boldsymbol{g}^{SAM}_1, \boldsymbol{g}^{SAM}_2, \boldsymbol{g}^{SAM}_3$.
5: - Gradient aggregation:
$\bar{\boldsymbol{g}} = \mathrm{PCGrad}(\boldsymbol{g}^{SAM}_1, \boldsymbol{g}^{SAM}_2, \boldsymbol{g}^{SAM}_3)$
6: - Update model: $\boldsymbol{w} = \boldsymbol{w} - \eta \bar{\boldsymbol{g}}$
7: **end for**
8: Add data D_t to buffer B

4.4 Discussion

We have proposed a direct application of SAM on a simple memory-based baseline which can boost CL performance due to the flatness of the loss landscape. We have further hypothesized the potential risk of SAM gradient conflict and proposed to apply one gradient aggregation strategy to tackle this risk. Regarding the direct approach, it is worth noting that Deng et al. [7] also investigated the impact of the loss landscape's flatness in the context of CL, and applied SAM to their method. However, they focused on GPM, a gradient-based method, and reported accuracy in the task-incremental setting (TIL). Therefore, from their work, it is not clear to justify how beneficial SAM is for a more general memory-based method under a more realistic evaluation scenario, class-incremental learning (CIL). Meanwhile, we provide experimental results for both mentioned scenarios.

Regarding the application of PCGrad, we assume that the model from the most recent task, $\boldsymbol{w}^*_{t-1}$, has been well-trained to effectively resolve SAM gradient conflict among past classes. In other words, $\boldsymbol{w}^*_{t-1}$ is an optimal flat solution for all tasks so far, not favoring any particular task. Under this assumption, we only need to mitigate gradient disagreement between the loss of current data and the loss(es) of buffer data. This is practically beneficial as if we individually consider all $t-1$ old tasks, represented by $t-1$ terms in the total loss, the calculation of each task's gradient will cause great computational and memory complexity.

In terms of time limitations, our method takes around 3.5 times as much as a memory-replay baseline does (no SAM and no PCGrad). In particular, similar to SAM, our method first performs one forward-backward pass for each term in the loss function, then another forward-backward pass to compute their SAM gradients, and finally aggregates them. Regarding space complexity, for each muni-batch update, our method requires three more times space to store gradients of the three loss terms.

5 EXPERIMENTS

Datasets. In our experimentation, we apply the proposed method to a range of widely used continual learning datasets, including sequential CIFAR-10, sequential Tiny ImageNet, Permuted MNIST, and Rotated MNIST. To provide some context, CIFAR-10 includes 10 classes with 5000 training and 1000 test examples each, while Tiny ImageNet, a subset of ImageNet, comprises 200 classes with 500 training examples per class. Sequential-CIFAR-10 (S-CIFAR-10) contains 5 sequential tasks, each involving 2 classes. Sequential-Tiny-ImageNet (S-Tiny-ImageNet) divides the 100 classes of Tiny ImageNet into 20 tasks, each consisting of 5 classes. Permuted MNIST (P-MNIST) and Rotated MNIST (R-MNIST) consist of 20 subsequent tasks; the former shuffles pixel positions using random permutations, while the latter rotates input images by random angles within the $[0; \pi)$ range.

Architectures. We utilize a standard ResNet18 architecture for CIFAR-10 and Tiny ImageNet without pretraining; A fully connected network with two hidden layers for the Permuted and Rotated MNIST. This is consistent with [2]. All networks use the ReLU activation function, and the classification loss is cross-entropy loss on the softmax layer.

Experimental settings. S-CIFAR-10 and S-Tiny-ImageNet are used in CIL and TIL settings, while P-MNIST and R-MNIST are for DIL. All experiments use Stochastic Gradient Descent (SGD) to optimize the model. We set the number of epochs to 50 and 100 for the first two datasets, respectively, and 1 for the other two. In each mini-batch update, we randomly sample from the buffer with the number of samples equal to that of the current mini-batch. The buffer is updated using Reservoir Sampling [6] at each of each task for S-CIFAR-10 and S-Tiny-ImageNet, and at each of each mini-batch update for the two MNIST datasets.

Evaluation protocol. We adopt the commonly used metric to measure the overall performance of a CL method, Average Accuracy: $Avg - Acc = \frac{1}{T}\sum_{i=1}^{T} A_{T,i}$, where T is the total number of tasks and $A_{T,i}$ is the accuracy of task i after the model has learned task T.

Baselines. We compare SGAM-CL with several baselines in three main CL approaches, including: Regularization-based: oEWC [13], SI [29], LwF [15]; Architecture-based: PNN [23]; and Memory-based: ER [6], GEM [16], A-GEM [5], iCaRL [22], HAL [4], DER [2]. We include sequential training without any CL techniques (SGD) and joint training with data from all tasks (JOINT) as lower-bound and upper-bound, respectively. We also compare SGAM-CL with F2M and CPR, two flat-seeking methods used in CL. For a fair comparison, we also incorporate them into the same baseline that SGAM-CL uses (Eq.6), namely ER-F2M and ER-CPR.

Hyper-parameters. We use the reported results from [2] as we follow their experimental settings. For ER-F2M [26] and ER-CPR [3], we tune the number of layers to which F2M injects noises in $\{2, 3, 4\}$, and the bound of these noises in $\{0.001, 0.01\}$, and the weight term enforcing flatness used by CPR in $\{0.02, 0.05, 0.1, 0.2, 0.5\}$. For SGAM-CL, we tune the radius perturbation ρ in $\{0.01, 0.05, 0.1, 0.5, 1.0\}$. After tuning, for ER-F2M, the number of layers is 3 and the bound is 0.001. For ER-CPR, the weight term is 0.1. For SGAM-CL, ρ is 0.1.

5.1 Experimental analyses

Main results. Table 1 shows the average accuracy of our proposed method and baselines on standard CL benchmarks. For the different values of buffer size, our algorithm SGAM-CL which aims to find the common flat optimal regions for the proposed terms in the objective function obtains the best results. Notably, on S-CIFAR-10 and S-Tiny-ImageNet, SGAM-CL outperforms the best baseline, by a large margin with a buffer size of 200 and 500 under the CIL setting. With S-CIFAR-10 under TIL, the gap is less significant as the model only needs to infer the correct class between two classes (this dataset has two classes per task), but this trend is different with the more challenging dataset S-Tiny-ImageNet: SGAM-CL still boosts TIL performance considerably. Under the DIL scenario, we still obtain better results across all settings than DER, DER++, and two flat versions of DER++, and surpass other baselines significantly. This large can be attributed to the special buffer update scheme of DER that saves a sample's logits (pre-softmax) with its hard label to the buffer along the training trajectory. By seeking flat minima, we further improve the performance.

As a side note, regularization-based methods, oEWC, SI, and LwF, which do not replay old data, perform poorly, especially under

Table 1: Classification results for standard CL benchmarks. '-' means the experiments were not feasible due to: intractable training time (GEM, HAL on S-Tiny-ImageNet), no access to task-id when testing (PNN under CIL), incompatibility under DIL (ICaRL, PNN, LwF). Bold font denotes best results; Italic font denotes results of CPR and F2M.

Buffer	Method	S-CIFAR-10		S-Tiny-ImageNet		P-MNIST	R-MNIST
		Class-IL(↑)	Task-IL(↑)	Class-IL(↑)	Task-IL(↑)	Domain-IL(↑)	Domain-IL(↑)
-	JOINT	92.20±0.15	98.31±0.12	59.99±0.19	82.04±0.10	94.33±0.17	95.76±0.04
	SGD	19.62±0.05	61.02±3.33	7.92±0.26	18.31±0.68	40.70±2.33	67.66±8.53
-	oEWC [13]	19.49±0.12	68.29±3.92	7.58±0.10	19.20±0.31	**75.79±2.25**	**77.35±5.77**
	SI [29]	19.48±0.17	68.05±5.91	6.58±0.31	36.32±0.13	65.86±1.57	71.91±.83
	LwF [15]	**19.61±0.05**	63.29±2.35	**8.46±0.22**	15.85±0.58	-	-
	PNN [23]	-	**95.13±0.72**	-	**67.84±0.29**	-	-
200	ER [6]	44.79±1.86	91.19±0.94	8.49±0.16	38.17±2.00	72.37±0.87	85.01±1.90
	GEM [16]	25.54±0.76	90.44±0.94	-	-	66.93±1.25	80.80±1.15
	A-GEM [5]	20.04±0.34	83.88±1.49	8.07±0.08	22.77±0.03	66.42±4.00	81.91±0.76
	iCaRL [22]	49.02±3.20	88.99±2.13	7.53±0.79	28.19±1.47	-	-
	HAL [4]	32.36±2.70	82.51±3.20	-	-	74.15±1.65	84.02±0.98
	DER [2]	61.93±1.79	91.40±0.92	11.87±0.78	40.22±0.67	81.74±1.07	90.04±2.61
	DER++ [2]	64.88±1.17	91.92±0.60	10.96±1.17	40.87±1.16	83.58±0.59	90.43±1.87
	F2M-DER++	65.36±1.76	92.25±0.58	11.66±0.93	40.30±1.56	83.76±0.27	90.66±0.70
	CPR-DER++	66.17±1.12	91.78±0.25	11.54±0.47	41.00±0.21	83.69±0.79	90.49±1.40
	SGAM-CL (Ours)	**70.44±0.18**	**93.95±0.15**	**21.02±0.40**	**56.41±1.75**	**84.70±0.96**	**90.78±0.27**
500	ER[6]	57.74±0.27	93.61±0.27	9.99±0.29	48.64±0.46	80.60±0.86	88.91±1.44
	GEM [16]	26.20±1.26	92.16±0.69	-	-	76.88±0.52	81.15±1.98
	A-GEM [5]	22.67±0.57	89.48±1.45	8.06±0.04	25.33±0.49	67.56±1.28	80.31±6.29
	iCaRL [22]	47.55±3.95	88.22±2.62	9.38±1.53	31.55±3.27	-	-
	HAL [4]	41.79±4.46	84.54±2.36	-	-	80.13±0.49	85.00±0.96
	DER [2]	70.51±1.67	93.40±0.39	17.75±1.14	51.78±0.88	87.29±0.46	92.24±1.12
	DER++ [2]	72.70±1.36	93.88±0.50	19.38±1.41	51.91±0.68	88.21±0.39	92.77±1.05
	F2M-DER++	72.85±1.32	93.95±0.72	19.40±0.82	51.48±0.77	88.46±0.36	92.90±0.17
	CPR-DER++	73.21±1.43	94.11±0.65	19.04±0.74	53.46±0.42	88.65±0.25	92.81±1.03
	SGAM-CL (Ours)	**76.26±1.02**	**95.25±0.41**	**26.38±0.12**	**62.59±0.50**	**88.94±0.69**	**93.03±0.17**
5120	ER [6]	82.47±0.52	96.98±0.17	27.40±0.31	67.29±0.23	89.90±0.13	93.45±0.56
	GEM [16]	25.26±3.46	95.55±0.02	-	-	87.42±0.95	88.57±0.40
	A-GEM [5]	21.99±2.29	90.10±2.09	7.96±0.13	26.22±0.65	73.32±1.12	80.18±5.52
	iCaRL [22]	55.07±1.55	92.23±0.84	14.08±1.92	40.83±3.11	-	-
	HAL [4]	59.12±4.41	88.51±3.32	-	-	89.20±0.14	91.17±0.31
	DER [2]	83.81±0.33	95.43±0.33	36.73±0.64	69.50±0.26	91.66±0.11	94.14±0.31
	DER++ [2]	85.24±0.49	96.12±0.21	39.02±0.97	69.84±0.63	92.26±0.17	94.65±0.33
	F2M-DER++	85.38±0.53	96.19±0.75	39.34±0.38	69.41±0.99	92.19±0.10	94.43±0.15
	CPR-DER++	86.12±0.42	96.11±0.38	39.45±0.87	69.88±0.54	92.11±0.11	94.53±0.11
	SGAM-CL (Ours)	**86.58±0.46**	**97.30±0.20**	**42.19±0.52**	**73.70±0.44**	**92.26±0.10**	**94.93±0.27**

CIL, compared to exemplar-based ones. This suggests that preserving knowledge of a past task through constraining its important parameters is not sufficient.

To sum up, being aware of sharpness for each term in the loss function, SGAM-CL improves DER++, the exemplar replay baseline using the same objective (Eq. 6), by nearly 6% and 10% on S-CIFAR-10 and S-Tiny-ImageNet under a limited buffer size, respectively.

SGAM - A better flat seeking optimizer for CL. Additionally, in comparison with CPR-DER++ and F2M-DER++, whose results are underlined in Table 1, SGAM-CL clearly shows its superiority over the other two across all settings, especially under the challenging S-Tiny-ImageNet dataset. Meanwhile, the improvement F2M and CPR bring to DER++ is relatively minor compared to that of SGAM, and in some cases, they even degrade DER++, e.g. P-MNIST and R-MNIST with buffer 5120. Regarding F2M, we follow their experiments to consider 2 random times to ensure feasible training time. This probably results in poor performance of F2M-DER++ compared to SGAM-CL in practice. Concerning CPR, it generally can work better than F2M under CIL but still falls behind SGAM, especially with a limited buffer.

5.2 Ablation Study

Table 2: Ablation studies of SGAM-CL on S-CIFAR-10, 200 buffer size. (-) means ablated, (+) means used.

SAM	Grad Aggr	Class-IL(↑)	Task-IL(↑)
-	-	65.13±0.86	92.23±0.70
-	+	66.82±2.85	92.78±0.41
+	-	69.64±0.06	93.79±0.02
+	+	**70.44±0.18**	**93.95±0.15**

SAM and Gradient Aggregation. Table 2 presents the effect of different components in our proposed method, SAM and Gradient

Aggregation (PCGrad), using S-CIFAR-10 with 200 buffer size. First, it can be observed that the improvement brought to the baseline by SAM is higher by PCGrad. This could be because gradient conflict is not a big problem in the loss terms of the baseline, which seems in line with existing CL methods as they often neglect gradient conflict when minimizing the empirical loss. Nevertheless, when combining both SAM and PCGrad, the model performance is boosted the most.

SGAM-CL and SAM-CL. In Section 4.2, we have conjectured that there might exist a conflict between the SAM gradient of each loss term, and thus the solution found by SAM-CL may not be flat for both current and old data. Therefore, a step of resolving gradient conflict is added before the final model update step can improve the solution. We will empirically verify this conjecture and show that SGAM-CL does improve SAM-CL.

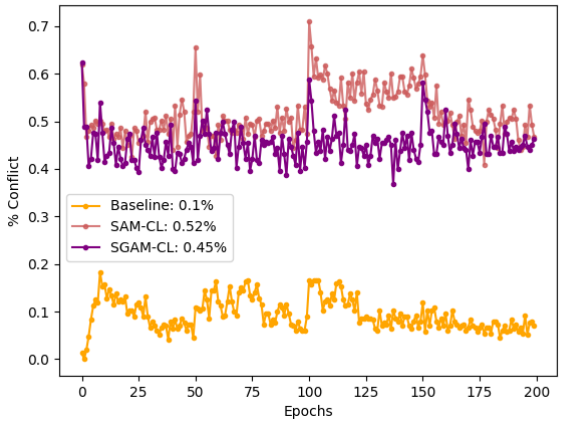

Figure 1: The ratio of conflict in one epoch along 5-task-training of S-CIFAR-10, and the average ratio of Baseline, SAM-CL and SGAM-CL.

First, let us define the *conflict number* in one training epoch: the number of mini-batch updates that have negative cosine similarity between the gradients of loss term on current data and one on buffer data. Then *conflict ratio* equals to the ratio between *conflict number* and the total number of updates in one epoch. We plot this ratio along the 5-task training on S-CIFAR-10 of the baseline, SAM-CL, and SGAM-CL in Figure 1. It can be seen that the *conflict ratio* of SAM-CL is consistently higher than that of the baseline, showing that there is more conflict in SAM-CL than in Baseline (0.52% vs 0.1%). After aggregating gradients, SGAM-CL does decrease the conflict (0.45%), but in comparison with Baseline, there still exists a large gap. Despite this, SGAM-CL still outperforms Baseline, as was shown in Table 2. Therefore, it is interesting to investigate if further closing this gap will result in a better SGAM-CL, which we leave for future work.

Second, we show that the solution obtained by SAM-CL is less robust against small parameter perturbations than SGAM-CL. Following [2], we add zero-mean Gaussian noise with increasing variance to the optimal parameters learned by the two methods, then visualize the average training loss and accuracy of all tasks, as can be seen from Figure 2. Clearly, SGAM-CL shows stronger robustness as its training loss is always smaller than that of SAM-CL, and similarly with the training accuracy, except for the final perturbation.

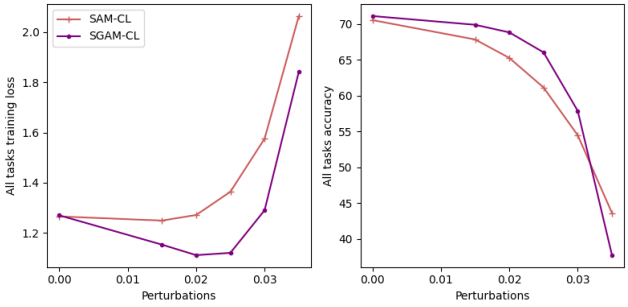

Figure 2: Loss (left) and Accuracy (right) changes w.r.t weight perturbation of SGAM-CL and SAM-CL.

Third, we present the full average accuracy of SAM-CL and SGAM-CL for S-CIFAR-10 and S-Tiny-ImageNet with various buffer sizes in Table 3. These consistent improvements of SGAM-CL over SAM-CL further prove that being aware of gradient congruence does benefit sharpness-aware minimization in memory-based CL methods.

Table 3: Comparison between SGAM-CL and SAM-CL on S-CIFAR-10 and S-Tiny-Imagenet.

Dataset	Buffer size	SAM-CL		SGAM-CL	
		Class-IL(↑)	Task-IL(↑)	Class-IL(↑)	Task-IL(↑)
S-CIFAR-10	200	69.64	93.79	**70.44**	**93.95**
	500	74.46	95.12	**76.26**	**95.25**
	5120	85.57	96.94	**86.58**	**97.30**
S-Tiny-ImageNet	200	20.26	53.09	**21.02**	**56.41**
	500	25.27	60.30	**26.38**	**62.59**
	5120	40.63	72.56	**42.19**	**73.70**

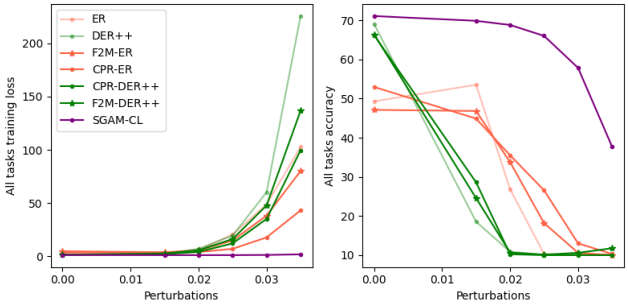

Figure 3: Loss (left) and Accuracy (right) changes w.r.t weight perturbation of SAM-CL and several baselines.

SGAM-CL's robustness vs other baselines. Finally, we visualize model sensitivity to local perturbations of the proposed method and other baselines in Figure 3. It can be seen that among methods that explicitly seek flat minima (F2M-ER, CPR-ER, F2M-DER++, CPR-DER++), SGAM-CL possesses a higher tolerance to perturbations, showing that it has converged to a flatter local minima than the others have. This positive correlation between flat minima and CL performance is in line with previous observations in [2, 3].

6 CONCLUSION

We have proposed a method that can significantly boost the performance of a simple memory-based baseline by exploiting a well-known approach: flatness enforcement. In particular, we leverage the effectiveness of SAM to make the model less overfitting, especially on a limited buffer. Moreover, we present a hypothesis on potential gradient conflict between loss terms corresponding to new and old data, from which a gradient-aware version of SAM is proposed to further improve CL performance. Extensive experiments and analyses confirm our hypothesis and the effectiveness of our method. Despite these promising results, better ones could be obtained for example by considering variations of SAM and gradient aggregation methods, or further studying the conflict gap between SGAM-CL and the baseline counterpart. The latter could be an interesting direction for future work.

ACKNOWLEDGMENTS

This research was funded by Vingroup Innovation Foundation (VINIF) under project code VINIF.2021.ThS.BK.03.

REFERENCES

[1] Lorenzo Bonicelli, Matteo Boschini, Angelo Porrello, Concetto Spampinato, and Simone Calderara. 2022. On the effectiveness of lipschitz-driven rehearsal in continual learning. *Advances in Neural Information Processing Systems* 35 (2022), 31886–31901.

[2] Pietro Buzzega, Matteo Boschini, Angelo Porrello, Davide Abati, and Simone Calderara. 2020. Dark experience for general continual learning: a strong, simple baseline. *Advances in neural information processing systems* 33 (2020), 15920–15930.

[3] Sungmin Cha, Hsiang Hsu, Taebaek Hwang, Flavio P Calmon, and Taesup Moon. 2020. CPR: classifier-projection regularization for continual learning. *arXiv preprint arXiv:2006.07326* (2020).

[4] Arslan Chaudhry, Albert Gordo, Puneet Dokania, Philip Torr, and David Lopez-Paz. 2021. Using hindsight to anchor past knowledge in continual learning. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 35. 6993–7001.

[5] Arslan Chaudhry, Marc'Aurelio Ranzato, Marcus Rohrbach, and Mohamed Elhoseiny. 2018. Efficient lifelong learning with a-gem. *arXiv preprint arXiv:1812.00420* (2018).

[6] Arslan Chaudhry, Marcus Rohrbach, Mohamed Elhoseiny, Thalaiyasingam Ajanthan, Puneet K Dokania, Philip HS Torr, and Marc'Aurelio Ranzato. 2019. On tiny episodic memories in continual learning. *arXiv preprint arXiv:1902.10486* (2019).

[7] Danruo Deng, Guangyong Chen, Jianye Hao, Qiong Wang, and Pheng-Ann Heng. 2021. Flattening sharpness for dynamic gradient projection memory benefits continual learning. *Advances in Neural Information Processing Systems* 34 (2021), 18710–18721.

[8] Pierre Foret, Ariel Kleiner, Hossein Mobahi, and Behnam Neyshabur. 2020. Sharpness-aware minimization for efficiently improving generalization. *arXiv preprint arXiv:2010.01412* (2020).

[9] Pierre Foret, Ariel Kleiner, Hossein Mobahi, and Behnam Neyshabur. 2021. Sharpness-aware Minimization for Efficiently Improving Generalization. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net. https://openreview.net/forum?id=6Tm1mposlrM

[10] Robert M French. 1999. Catastrophic forgetting in connectionist networks. *Trends in cognitive sciences* 3, 4 (1999), 128–135.

[11] Saihui Hou, Xinyu Pan, Chen Change Loy, Zilei Wang, and Dahua Lin. 2019. Learning a unified classifier incrementally via rebalancing. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 831–839.

[12] Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. 2016. On large-batch training for deep learning: Generalization gap and sharp minima. *arXiv preprint arXiv:1609.04836* (2016).

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

[14] Lilly Kumari, Shengjie Wang, Tianyi Zhou, and Jeff A Bilmes. 2022. Retrospective adversarial replay for continual learning. *Advances in Neural Information Processing Systems* 35 (2022), 28530–28544.

[15] Zhizhong Li and Derek Hoiem. 2017. Learning without forgetting. *IEEE transactions on pattern analysis and machine intelligence* 40, 12 (2017), 2935–2947.

[16] David Lopez-Paz and Marc'Aurelio Ranzato. 2017. Gradient episodic memory for continual learning. *Advances in neural information processing systems* 30 (2017).

[17] Kaifeng Lyu, Zhiyuan Li, and Sanjeev Arora. 2022. Understanding the generalization benefit of normalization layers: Sharpness reduction. *Advances in Neural Information Processing Systems* 35 (2022), 34689–34708.

[18] Marc Masana, Xialei Liu, Bartłomiej Twardowski, Mikel Menta, Andrew D Bagdanov, and Joost Van De Weijer. 2022. Class-incremental learning: survey and performance evaluation on image classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 45, 5 (2022), 5513–5533.

[19] Seyed Iman Mirzadeh, Mehrdad Farajtabar, Dilan Gorur, Razvan Pascanu, and Hassan Ghasemzadeh. 2020. Linear mode connectivity in multitask and continual learning. *arXiv preprint arXiv:2010.04495* (2020).

[20] Seyed Iman Mirzadeh, Mehrdad Farajtabar, Razvan Pascanu, and Hassan Ghasemzadeh. 2020. Understanding the role of training regimes in continual learning. *Advances in Neural Information Processing Systems* 33 (2020), 7308–7320.

[21] Henning Petzka, Michael Kamp, Linara Adilova, Cristian Sminchisescu, and Mario Boley. 2021. Relative flatness and generalization. *Advances in neural information processing systems* 34 (2021), 18420–18432.

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

[23] Andrei A Rusu, Neil C Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. 2016. Progressive neural networks. *arXiv preprint arXiv:1606.04671* (2016).

[24] Gobinda Saha, Isha Garg, and Kaushik Roy. 2021. Gradient projection memory for continual learning. *arXiv preprint arXiv:2103.09762* (2021).

[25] Joan Serra, Didac Suris, Marius Miron, and Alexandros Karatzoglou. 2018. Overcoming Catastrophic Forgetting with Hard Attention to the Task. In *Proceedings of the 35th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 80)*, Jennifer Dy and Andreas Krause (Eds.). PMLR, 4548–4557. https://proceedings.mlr.press/v80/serra18a.html

[26] Guangyuan Shi, Jiaxin Chen, Wenlong Zhang, Li-Ming Zhan, and Xiao-Ming Wu. 2021. Overcoming catastrophic forgetting in incremental few-shot learning by finding flat minima. *Advances in neural information processing systems* 34 (2021), 6747–6761.

[27] Hanul Shin, Jung Kwon Lee, Jaehong Kim, and Jiwon Kim. 2017. Continual Learning with Deep Generative Replay. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett (Eds.). 2990–2999. https://proceedings.neurips.cc/paper/2017/hash/0efbe98067c6c73dba1250d2beaa81f9-Abstract.html

[28] Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. 2020. Gradient surgery for multi-task learning. *Advances in Neural Information Processing Systems* 33 (2020), 5824–5836.

[29] Friedemann Zenke, Ben Poole, and Surya Ganguli. 2017. Continual Learning through Synaptic Intelligence. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70* (Sydney, NSW, Australia) *(ICML'17)*. JMLR.org, 3987–3995.

[30] Yang Zhao, Hao Zhang, and Xiuyuan Hu. 2022. Penalizing Gradient Norm for Efficiently Improving Generalization in Deep Learning. In *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA (Proceedings of Machine Learning Research, Vol. 162)*, Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvári, Gang Niu, and Sivan Sabato (Eds.). PMLR, 26982–26992. https://proceedings.mlr.press/v162/zhao22i.html