CNNs Sparsification and Expansion for Continual Learning

Basile Tousside[1] [a], Jörg Frochte[1] [b] and Tobias Meisen[2] [c]

[1]*Bochum University of Applied Science, Kettwiger Str. 20, 42579 Heiligenhaus, Germany*

[2]*Bergische Universität Wupeprtal, Rainer-Gruenter-Straße 21, 42119 Wuppertal, Germany*

Keywords: Deep Learning, Convolutional Neural Networks, Continual Learning, Catastrophic Forgetting.

Abstract: Learning multiple sequentially arriving tasks without forgetting previous knowledge, known as Continual Learning (CL), remains a long-standing challenge for neural networks. Most existing CL methods rely on data replay. However, they are not applicable when past data is unavailable or is not allowed to be synthetically generated. To address this challenge, we propose Sparification and Expansion-based Continual Learning (SECL). SECL avoids forgetting of previous tasks by ensuring the stability of the CNN via a stability regularization term, which prevents filters detected as important for past tasks to deviate too much when learning a new task. On top of that, SECL makes the network plastic via a plasticity regularization term that leverage the over-parameterization of CNNs to efficiently sparsify the network and tunes unimportant filters making them relevant for future tasks. Also, SECL enhances the plasticity of the network through a simple but effective heuristic mechanism that automatically decides when and where (at which layers) to expand the network. Experiments on popular CL vision benchmarks show that SECL leads to significant improvements over state-of-the-art method in terms of overall CL performance, as measured by classification accuracy as well as in terms of avoiding catastrophic forgetting.

1 INTRODUCTION

In recent years, convolutional neural networks (CNNs) trained on massive amounts of data have achieved impressive results in diverse domains, such as medical imaging (Hering et al., 2022), object detection (Sharma et al., 2022) or face recognition (Schofield et al., 2023), often surpassing human capabilities. However, they typically rely on the strong assumptions that (i) all data are available at the same time and (ii) the data are independently and identically distributed (IID). These assumptions are violated in many practical applications that either deal with non-stationary data distributions or involve scenarios where data is not available at the same time but is generated sequentially. For example, a robot trained in a factory with a set of standard object recognition capabilities and deployed in a home or building may need to adapt to new areas and even solve new location-specific object recognition tasks. Another example is data privacy, where previous data must often be deleted after a certain period of time to protect user privacy, resulting in data from previous tasks not being available anymore when learning new tasks.

In such scenarios, if there is a domain shift between tasks and the model learner cannot access all tasks data at once, therefore focusing its learning exclusively on the data of the current task, a drop in performance occurs on the old tasks, a phenomenon referred to as *catastrophic forgetting* (Hayes et al., 2020). Overcoming catastrophic forgetting while limiting computational cost and memory requirements is the focus of *Continual learning* (CL), also known as lifelong- or sequential learning.

At the heart of catastrophic forgetting is the stability-plasticity dilemma (Grossberg, 1987), where a model balances between integrating new knowledge (plasticity) and preserving previously acquired ones (stability). If the model is naively trained on each new task without any measures against forgetting, it will be plastic but not stable, meaning that it can learn fast, but also forgets quickly. On the other hand, if too much attention is paid on stabilizing the previously learned knowledge, it will lack plasticity, i.e., sufficient capacity to learn new tasks. Lose of balance between stability and plasticity would deteriorate the performance of the continual learner model.

[a] https://orcid.org/0000-0002-9332-5060
[b] https://orcid.org/0000-0002-5908-5649
[c] https://orcid.org/0000-0002-1969-559X

Tousside, B., Frochte, J. and Meisen, T.
CNNs Sparsification and Expansion for Continual Learning.
DOI: 10.5220/0012314000003636
Paper published under CC license (CC BY-NC-ND 4.0)
In *Proceedings of the 16th International Conference on Agents and Artificial Intelligence (ICAART 2024) - Volume 2*, pages 110-120
ISBN: 978-989-758-680-4; ISSN: 2184-433X
Proceedings Copyright © 2024 by SCITEPRESS – Science and Technology Publications, Lda.

Previous work has addressed the stability aspect by identifying weights that were important in learning old tasks and either freezing them or constraining them not to change significantly (Kirkpatrick et al., 2017; Smith et al., 2023). Plasticity, on the other hand, is typically handled by expanding the network so that the additional capacity can focus on assimilating new knowledge (Ostapenko et al., 2019; Gurbuz and Dovrolis, 2022). However, many expansion-based CL methods introduce new filters/neurons without optimizing the use of existing capacity (Yoon et al., 2017; Yan et al., 2021).

In this paper, we tackle the problem from a novel angle that leverages sparsification to seek a better balance between stability and plasticity. Specifically, we propose to sparingly use the available network capacity before considering network expansion. Taking into account that neural networks are overparameterized, so that there is often more capacity than needed to learn a given task (Sankararaman et al., 2020), our Sparification and Expansion-based Continual Learning (SECL) method aims to regularize the parameters update so that the sparsity of the weights is reinforced during learning.

After learning a task t, we identify two categories of filters: (i) those that were crucial for learning that task, which we constrain to remain stable via a stability regularizer, and (ii) those that remained unused due to sparsification, which we reinitialize to learn future tasks. For scenarios with a limited number of tasks, such a sparsification strategy can effectively address the stability-plasticity dilemma. However, when faced with a continually increasing number of tasks, this approach becomes less effective as the network capacity eventually saturates after assimilating a specific number of tasks.

Consequently, there are two initial situations in the training process of a new task t: (i) the capacity limit has not yet been reached, the available filters can be used to learn this task, (ii) the capacity limit has been reached, further learning inevitably leads to forgetting, hence the capacity must be expanded. CL approaches based on network expansion often grow the network every time a new task arrives (Rusu et al., 2016; Yan et al., 2021), whereas our method incorporates a strong heuristic to decide when to expand the network. Moreover, typical CL expansion methods often add new filters either at every layer or only at higher (classification) layers. In contrast, our approach allows us to add capacity only where it is needed.

For the remainder of the paper, we will outline four main contributions of our SECL methods:

- SECL uses sparsity to improve CL. Instead of the common practice of using a dense architecture, we reinforce a sparse network architecture during the learning of each task, which ensures the plasticity of the network with respect to future tasks.
- Unlike dense architectures, sparse architectures allow us to rewire connections. This reactivates unused filters, allowing new knowledge to be encoded.
- SECL preserves previously acquired knowledge by penalizing deviations in key parameters for past tasks.
- Finally, SECL provides a strong heuristic for deciding when and where to expand a continual learning network.

2 RELATED WORK

Continual learning has received considerable attention in recent years. In this section, we provide a brief survey of current state of the art approaches to address CL. These approaches are typically categorized in 3 mains groups: expansion-based methods, regularization-based methods, and memory-based methods. We will elaborate more on the first group since the work presented in this paper fall into this category. For an exhaustive survey, see (Parisi et al., 2019; Zhou et al., 2023).

Expansion-Based Methods. This first group deals with continual learning by dynamically increasing the capacity of the network to reduce interference between new tasks and old ones (Hung et al., 2019; Yan et al., 2021; Douillard et al., 2022; Wang et al., 2022a). The key idea is to dedicate different subsets of the final network to each task. One notable work is Progress & Compress (P&C) (Schwarz et al., 2018), which extends the network architecture for new tasks and freezes the parameters learned for old tasks. Thus, new capacities are added to learn new tasks while previous knowledge is preserved by the frozen weights. Similarly, Dynamically Expandable Representation (DER) (Yan et al., 2021) freeze the previously learned representation and augment it with additional feature dimensions from a new learnable feature extractor. Both approaches, however, focus on the stability of the network, thereby largely neglecting its plasticity for learning new tasks. Our work addresses this shortcoming via the introduced plasticity regularizer. Another notable difference in our approach is that prior methods typically work at a parameter level (individual parameter of a convolution filter for example), whereas ours works on a group of parameters (the entire filter).

Others expansion-based approaches promote a sparse architecture similar to ours, but have some limitations. For instance, CPG (Hung et al., 2019) iteratively removes a small portion of the weights and retrains the remaining weights to restore the original performance. The procedure terminates when a predefined accuracy target is reached, for which the difficulty of the task must be known in advance. SparCL (Wang et al., 2022b) uses a task-aware dynamic masking strategy to keep only important weights for both the current and past tasks, with special consideration during task transitions. However, the method assumes a rehearsal buffer to be available throughout the continual learning process. SNCL (Yan et al., 2022) employs variational Bayesian sparsity priors on the activations of the neurons in all layers to only activate and select sparse neurons for learning current and past tasks at any stage. However, they use a full experience replay to provides effective supervision in learning the sparse activations of the neurons in different layers.

Regularization-Based Methods. This second group preserves prior knowledge by imposing penalties on significant deviations of parameters deemed crucial for earlier tasks (Maschler et al., 2021; Liu and Liu, 2022; Bonicelli et al., 2023; Smith et al., 2023). Thus, a key component is how to measure the importance of each network parameter in learning a given task. Notably, to determine weight importance, HAT (Serra et al., 2018) learns hard attention masks, MAS (Aljundi et al., 2018) evaluates the sensitivity of model outputs to inputs in an unsupervised manner, and PIGWM (Zhou et al., 2021) uses a full Hessian matrix.

Memory-Based Methods. The final group stores training examples for each old task (Rolnick et al., 2019; Douillard et al., 2022; Yoon et al., 2021; Caccia et al., 2022) or generates pseudo-examples of the old tasks (Lesort et al., 2019; Douillard et al., 2020; Sokar et al., 2021; Zeng et al., 2023). These stored or generated examples are afterwards used jointly when learning new tasks.

Methods in this last category are the oldest technique to deal with continual learning. However, concerns about data protection and privacy (storing or regenerating old task data) have led continual learning research focus on **regularized-** and **expansion**-based methods. The framework proposed in this paper integrates these two techniques into a robust CL approach capable of handling an infinite stream of tasks.

3 METHOD

Before describing our approach, we briefly introduce the formalism and notation used throughout the paper.

3.1 Notation

We consider T sequentially arriving and previously unknown, image classification tasks of supervised learning $\{\mathcal{T}^1, \cdots, \mathcal{T}^T\}$. Each task $\mathcal{T}^t$ consists of a set of classes to be learned together and has a training set $D^t_{train} = \{(x^t_s, y^t_s)\}^{m^t}_{s=1}$, where x^t_s is an input sample, y^t_s is its class label, and m^t is the number of training samples. Similarly, the validation and test data sets are denoted as D^t_{valid} and D^t_{test}, respectively. We further denote $l \in \{0, \cdots, L\}$ as a layer of the network and F_l as the number of its filters. Furthermore, $f_{l,i}\,(i \in \{1, \cdots, F_l\})$ denotes the i-th filter in layer l, $\theta_{l,i}$ its parameters and $\Omega_{l,i}$ its importance. Additionally, we denote the activation of filter $f_{l,i}$ by $a_{f_{l,i}}(x_s)$, where x_s is the input that generates this activation. Finally, $\theta^t = \{\theta^t_l\}_{1 \leq l \leq L}$ with $\theta^t_l = \{\theta^t_{l,i}\}_{1 \leq i \leq F_l}$ denotes the network parameters at task t.

3.2 Preventing Forgetting

Filter Importance $\overline{\Omega^{t-1}_{l,i}}$ at Learning $\mathcal{T}^{t-1}$. To tackle the catastrophic forgetting challenge described in Section 1, we use a strategy similar to Kirkpatrick et al. (Kirkpatrick et al., 2017), which consists of preventing weights that have been identified as important for past tasks from deviating too much when learning new tasks. However, while Kirkpatrick et al. use an approximation of the Fisher information matrix (which is computationally expensive and requires storing and manipulating large matrices) to measure weight importance, we use the post-activation averaged over all training samples. Moreover, while they ((Kirkpatrick et al., 2017)) operate on an individual parameter level, our method works on a group of parameters, i.e. the entire filter.

Specifically, we quantify the importance $\overline{\Omega^{t-1}_{l,i}}$ of a filter i in layer l at learning task $t-1$ as the average standard deviation of its post-activation value over all training samples at task $t-1$.

The standard deviation $\sigma^{t-1}_{f_{l,i}}$ of a filter's post-activation $a_{f_{l,i}}$ across all m^{t-1} training samples can be computed as:

$$\sigma^{t-1}_{f_{l,i}} = \sqrt{\frac{1}{m^{t-1}} \sum_{s=1}^{m^{t-1}} \left(a_{f_{l,i}}(x_s) - \hat{a}_{f_{l,i}}\right)^2}, \tag{1}$$

where $a_{f_{l,i}}(x_s)$ is the ReLU activation value of filter

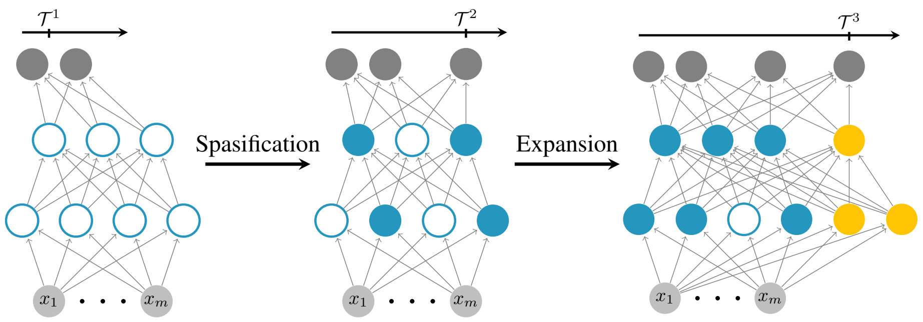


Figure 1: **Overview of SECL.** On the left, All neurons are unimportant at first. SECL trains on Task $\mathcal{T}^1$ with the sparsity regularizer as per Eq. 4, identifying neurons crucial for $\mathcal{T}^1$. In the center, during Task $\mathcal{T}^2$, the network trains with both sparsity and stability regularizers to preserve knowledge from the first task; neurons important for $\mathcal{T}^1$ are highlighted in blue. On the right, by the time task $\mathcal{T}^3$ arrives, almost all neurons have been marked as important for prior tasks, necessitating the addition of new capacity (depicted in yellow) to accommodate $\mathcal{T}^3$. At this stage, the network is further trained with the sparsity regularizer in order to use the additional capacity sparingly.

$f_{l,i}$ for the training sample $x_s \in \mathcal{D}^{t-1}_{train}$ and $\hat{a}_{f_{l,i}}$ is the average activation.

The intuition behind this method of quantifying filter importance is based on the observation that one of the reasons for the popularity of the ReLU activation functions is that it induces sparsity in activations. Hence, it is reasonable to assume that if the output activation value of the filter is low, then the feature detected by that filter (and thus the filter itself) is not important for learning the current task.

Filter Importance $\Omega^{t-1}_{l,i}$ at Learning $\mathcal{T}^1$ up to $\mathcal{T}^{t-1}$. When learning a task $\mathcal{T}^{t-1}$ with $t > 2$, we need to account for the importance of each filter for past tasks $\{\mathcal{T}^1, \cdots, \mathcal{T}^{t-2}\}$, to ensure stable performance on those tasks while adapting to new ones. Therefore, we update the importance of a filter $f_{l,i}$ in learning $\mathcal{T}^1$ to $\mathcal{T}^{t-1}$ as follows:

$$\Omega^{t-1}_{l,i} = \gamma\, \Omega^{t-2}_{l,i} + \overline{\Omega^{t-1}_{l,i}}\ , \qquad (2)$$

where $\Omega^{t-2}_{l,i}$ is the importance of $f_{l,i}$ in learning $\mathcal{T}^1$ up to $\mathcal{T}^{t-2}$, $\overline{\Omega^{t-1}_{l,i}}$ is the importance of $f_{l,i}$ in learning $\mathcal{T}^{t-1}$ and γ is a parameter weighting the importance of $f_{l,i}$ before and after learning $\mathcal{T}^{t-1}$.

Loss Function at Learning $\mathcal{T}^t$. When learning a new task $\mathcal{T}^t$ with $t > 1$, we can restrict filters that were important in learning past tasks $\{\mathcal{T}^1, \cdots, \mathcal{T}^{t-1}\}$ from changing too much, so that performance on those previous tasks is maintained. Remember that at this stage the importance of each filter in learning task $\mathcal{T}^1$ up to $\mathcal{T}^{t-1}$ is given by $\Omega^{t-1}_{l,i}$ as defined in equation 2.

The training objective in learning task t can then be formulated as:

$$L^t_{\mathsf{CL}}(\theta^t) = L^t_{\mathsf{CE}}(\theta^t) + \lambda_s \sum_{l=1}^{L} \sum_{i=1}^{F_l} \Omega^{t-1}_{l,i}\ \|\theta^t_{l,i} - \theta^{t-1}_{l,i}\|_2^2\ , \qquad (3)$$

where, L^t_{CL} is the continual learner loss during training at $\mathcal{T}^t$, L^t_{CE} is the cross-entropy loss and λ_s sets the influence of the stability penalty.

Equaton 3 ensures the stability of the network on previous tasks. The next challenge is to equip the network with a sparsification scheme that ensures optimal use of its capacity during the learning of each task.

3.3 Network Sparsification

Sparsity Regularizer at Learning $\mathcal{T}^t$. Even if a network has enough capacity to learn a given task, care must be taken to use it sparingly. Therefore, instead of directly expanding the network as traditional expansion-based CL methods do, our approach employs a sparsification mechanism to first use the capacity of the network in a sustainable manner.

Our sparification mechanism is inspired by model pruning and compression techniques. This appealing area of research aims to reduce the size of a model by removing redundant or less important parameters. Methods to achieve this generally consist of equipping the objective function with a sparsity regularizer acting on the network parameters. The most popular choice for such a regularizer is the *group sparsity* regularizer introduced by Yuan et al. (Yuan and Lin, 2006), which promotes feature sharing. Feature sharing is desired in the front layers of a CNN, which act as feature extractors. However, at deeper layers, which aim to differentiate between classes, it is more useful to use a sparsity regularizer that promotes feature discrimination. Such a regularizer has been proposed in the concepts of sparse group lasso or *exclu-*

sive sparsity in (Zhou et al., 2010). We use both the ideas of *group sparsity* and *exclusive sparsity* in our model, which allows us to write the sparsity regularizer (which we also denote as plasticity regularizer) as follows:

$$R(\theta^t) = \underbrace{\sum_{l=1}^{L}\sum_{i=1}^{F_l} \zeta_l \|\theta^t_{l,i}\|_2^2}_{\text{Group sparsity}} + \underbrace{\sum_{l=1}^{L}\sum_{i=1}^{F_l} \frac{(1-\zeta_l)}{2} \|\theta^t_{l,i}\|_1}_{\text{Exclusive sparsity}} , \tag{4}$$

where $\zeta_l = 1 - \frac{l}{L-1}$ sets the influence of both terms, giving more influence to *group sparsity* in front layers, while *exclusive sparsity* dominates in deeper layers.

Updated Loss Function at Learning $\mathcal{T}^t$. At this stage, our method summarizes as follows: during the first task ($t = 1$), the network is trained with the sparsity regularizer only (Eq. 4). Next, we compute the importance $\Omega^{t=1}_{l,i}$ of each filter in learning $\mathcal{T}^{t=1}$. By binarizing $\Omega^{t=1}_{l,i}$ based on a threshold, we can divide the parameter set θ into two parts, namely the parameters that were important for learning $\mathcal{T}^{t=1}$, denoted by θ_+, and the superfluous parameters that were not important, denoted by θ_-. When training further tasks $\mathcal{T}^t$ with $t > 1$, Eq. 3 and Eq. 4 can be combined to simultaneously (i) prevent forgetting past tasks (by constraining θ_+ not to change too much) and (ii) induce sparsity (over θ_-) such that only a minimal capacity of θ_- is used to learn $\mathcal{T}^t$.

The final training objective taking into account θ_+, θ_- and combining the stability regularizer (second part of Eq. 3) and the plasticity regularizer (Eq. 4) can then be framed as

$$\begin{aligned} L^t_{\mathsf{CL}}(\theta^t) = L^t_{\mathsf{CE}}(\theta^t) &+ \lambda_s \sum_{l=1}^{L} \sum_{\substack{i=1 \\ \forall\theta_{l,i}\in\theta_+^{t-1}}}^{F_l} \Omega^{t-1}_{l,i}\ \|\theta^t_{l,i} - \theta^{t-1}_{l,i}\|_2^2 \\ &+ \lambda_p \sum_{l=1}^{L} \sum_{\substack{i=1 \\ \forall\theta_{l,i}\in\theta_-^{t-1}}}^{F_l} \zeta_l \|\theta^t_{l,i}\|_2^2 \\ &+ \lambda_p \sum_{l=1}^{L} \sum_{\substack{i=1 \\ \forall\theta_{l,i}\in\theta_-^{t-1}}}^{F_l} \frac{(1-\zeta_l)}{2}\ \|\theta^t_{l,i}\|_1 , \end{aligned} \tag{5}$$

where λ_s and λ_p control the amount of stability and plasticity (sparsity) regularization applied to the model.

In order to minimize this training objective (Eq. 5), we follow a proximal gradient descent approach (Parikh and Boyd, 2014). In our context, proximal gradient descent (PGD) can be regarded as iteratively taking a gradient step with respect to $L^t_{\mathsf{CE}}(\theta^t)$ only, and then, from the resulting solution, applying the proximal operator of the stability and sparsity regularizer. In this setup, we apply the proximal operator at the end of each epoch and run the algorithm for a fixed number of epochs.

Algorithm 1 describes our learning process for sparification and catastrophic forgetting prevention.

Algorithm 1: Sparsity and Preventing Forgetting.

```
Input: $\mathcal{D} = (\mathcal{D}^1, \cdots, \mathcal{D}^T)$
Output: $\theta^T$
for $t = 1, \cdots, T$ do
    if $t = 1$ then
        Train the network parameters $\theta^{t=1}$ using
          Eq. 4;
    else
        Compute importance $\Omega^{t-1}_{l,i}$ of each filter
          in learing $\{\mathcal{T}^1, \cdots, \mathcal{T}^{t-1}\}$ using Eq. 2;
        Train the network parameters $\theta^t$ using
          Eq. 5, following a PGD optimization
          as proposed in (Parikh and Boyd,
          2014) ;
    end
end
```

3.4 Network Expansion

Even with a sparsification scheme as presented in the previous section, there may still come a point where network expansion becomes inevitable. This is particularly true if (i) the number of learning tasks is infinitely large, or (ii) there is a significant variability in task complexity, depending on the initial capacity of the network. In both scenarios, our method introduces additional filters into the network to capture the features needed to represent the new task. However, extending a network in a continual learning setting presents specific challenges. We categorized these challenges into three main groups, namely "scalability", "heuristics", and "dealing with additional parameters". The remainder of this section discusses each of these challenges in detail.

Scalability. The expansion of network capacity naturally leads to escalated training and storage costs. Many expansion-based continual learning methods (Yan et al., 2021), freeze the parameters related to old tasks to maintain performance, while adapting only the newly added parameters to the new task. This results in a model architecture whose complexity is proportional to the number of tasks, thus resulting in a highly redundant structure. To mitigate this significant drawback, we applied the sparsity-inducing regularization, as described in equation 4, to the filters

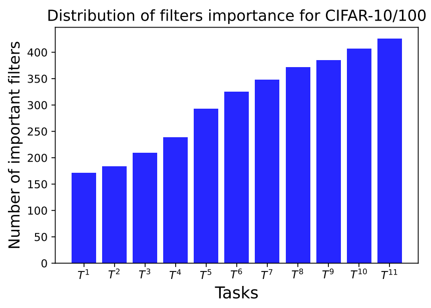


Figure 2: Evolution of important filters during the CIFAR-10/100 continual learning experiment: The plot shows the cumulative number of filters identified as important after the completion of each task.

introduced during expansion. Another feature of our method is the strategic addition of filters, specifically targeting layers where the number of unimportant filters falls below a predefined threshold. These two key characteristics allow a sustainable utilization of both the initial and the added capacity of the network, thus significantly improving the scalability of our method.

Heuristics. A major challenge in expansion-based continual learning approaches is to decide when to expand the network. At task t, if the previously learned parameter θ^{t-1} is able to effectively describe the new task, then network expansion, i.e., expansion of the parameter space, may not be necessary. On the other hand, if the previously learned features are not able to effectively describe the new task, then introducing additional filters becomes essential to accommodate the features needed to learn the new task. In view of this, there is a need for a robust heuristic capable to trigger network expansion when there is insufficient free capacity, or available filters, to assimilate the new task effectively.

Some works has proposed such heuristics to guide network expansion (Zhou et al., 2012; Yoon et al., 2017; Hung et al., 2019). However, their efficiency is often hampered by the fact that they require repeated passes through an iterative training process (Zhou et al., 2012), or by the introduction of hard requirements such as a threshold on either the loss (Yoon et al., 2017) or the accuracy (Hung et al., 2019), beyond which the network is expanded, which in both cases require knowing the task difficulty in advance. In constrast, we propose a simple yet effective heuristic to guide the network expansion. Our heuristic is based on a component of the objective function, namely the filters' importance vector Ω^t and thus does not require any additional assumptions about the task itself.

Specifically, when a predetermined percentage of the network's total filters are marked as important, we

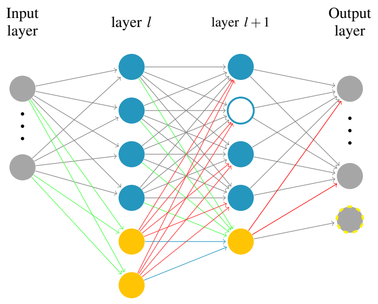


Figure 3: Schematic overview of the network expansion module: unimportant neurons are shown in white, important neurons in blue, and newly added neurons in yellow.

inspect the number of important filters within each layer. Additional filters are then introduced into layers where the available capacity is deemed to be insufficient. This allows our method to dynamically determine when to expand the network, and precisely how many filters should be added to each layer. Figure 2 provides a comprehensive view of the evolution of the filter importance vector $\{\Omega^t\}_{t=1}^{T}$ for the cifar-10/100 experiment. As evident from the figure, the more tasks are learned, the more filters are used and flagged as important.

Dealing with Additional Parameters. Another crucial aspect in network expansion is the handling of added filters or connections (in the context of a dense network). We will demonstrate our approach on a dense network (see Figure 3), where a "connection" represents a 2D kernel in a 3D convolution filter.

Consider the case where two consecutive layers, labeled l and $l+1$ are expanded by adding two and one neurons, respectively, as depicted in Figure 3. In this figure, the yellow-filled neurons symbolize the newly added neurons, while the yellow-dashed neuron represents the output neuron specifically added for the current task. This expansion process yields three distinct sets of parameter matrices:

- Matrix $\theta^{\text{expand},1} = [\theta^{in}_{l-1,l}; \theta^{in}_{l,l+1}]$ representing the incoming connections between each newly added neuron and the existing neurons in the layer below it. These connections are indicated in green in Figure 3.
- Matrix $\theta^{\text{expand},2} = [\theta^{in,out}_{l,l+1}]$ outlining the interconnections among the newly added neurons themselves, which are shown in blue in Figure 3.
- Matrix $\theta^{\text{expand},3} = [\theta^{out}_{l,l+1}; \theta^{out}_{l+1,l+2}]$ designating the outgoing connections from each added neuron to the existing neurons in the layer above it. These connections are highlighted in red in Figure 3.

Each of these matrices plays a distinct role in integrating the newly added neurons into the existing network architecture. Specifically, $\theta^{\mathsf{expand},1}$ (green connections) and $\theta^{\mathsf{expand},2}$ (blue connections) can be trained without any restrictions. However, care must be taken when dealing with $\theta^{\mathsf{expand},3}$ (red connections), otherwise, the information flowing to important neurons in layers $l+1$ and $l+2$ may change during the learning of the current task, resulting to catastrophic forgetting of previous knowledge.

To avoid this, we introduced an additional regularization term $\sum_{\theta_+^{\mathsf{expand},3}} \|\theta_{l,i}^t\|_2^2$ into the objective function. This term serves to strongly constrain the magnitude of $\theta^{\mathsf{expand},3}$ (red connections) specifically on neurons deemed important, thereby minimizing any potential interference with the retention of previously acquired knowledge. This strategy provides a balanced approach that allows for network expansion while maintaining the integrity of prior learning.

Accordingly, the learning objective changes when training with the network extension towards:

$$
\begin{aligned}
L_{\mathsf{CL}}^t(\theta^t) = L_{\mathsf{CE}}^t(\theta^t) &+ \lambda_s \sum_{l=1}^{L} \sum_{\substack{i=1 \\ \forall \theta_{l,i} \in \theta_+^{t-1}}}^{F_l} \Omega_{l,i}^{t-1} \, \|\theta_{l,i}^t - \theta_{l,i}^{t-1}\|_2^2 \\
&+ \lambda_p \sum_{l=1}^{L} \sum_{\substack{i=1 \\ \forall \theta_{l,i} \in \theta_-^{t-1}}}^{F_l} \zeta_l \|\theta_{l,i}^t\|_2^2 \\
&+ \lambda_p \sum_{l=1}^{L} \sum_{\substack{i=1 \\ \forall \theta_{l,i} \in \theta_-^{t-1}}}^{F_l} \frac{(1-\zeta_l)}{2} \, \|\theta_{l,i}^t\|_1 \\
&+ \lambda_e \sum_{l=1}^{L} \sum_{\substack{i=1 \\ \forall \theta_{l,i} \in \theta_+^{expand,3}}}^{F_l} \|\theta_{l,i}^t\|_2^2 \, ,
\end{aligned}
\tag{6}
$$

where λ_e is a hyperparameter that regulates the extent of expansion regularization. The optimization of this modified learning objective is carried out using a proximal gradient descent approach, already mentioned in Section 3.3. For an in-depth view of the proximal gradient descent method see (Parikh and Boyd, 2014).

Algorithm 2 presents the overall SECL framework.

4 EXPERIMENTS

Following, we present the empirical evaluation results comparing our work with various state-of-the-art continual learning baselines under different experimental settings. All experiments are performed using the pytorch library, and all results are averaged over five different random seeds.

Algorithm 2: Sparsification and Expansion-based CL (SECL) Algorithm.

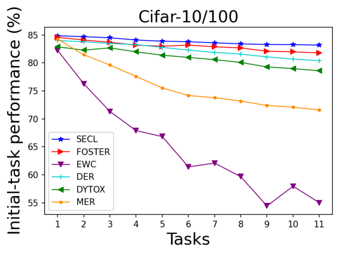
```
Input: Model parameters $\theta$,
       Task Dataset $\mathcal{D} = (\mathcal{D}^1, \cdots, \mathcal{D}^T)$,
       Threshold $\tau$
Output: $\theta^T$ : Model parameters after training
        on the last task $\mathcal{T}^T$
for $t = 1, \cdots, T$ do
    if $t = 1$ then
        Train the network parameters $\theta^{t=1}$ using
          the sparsity regularizer only (Eq. 4);
    else
        Compute importance $\Omega_{l,i}^{t-1}$ of each filter
          in learning tasks up to $\mathcal{T}^{t-1}$ (Eq. 2) ;
        if number of important filters $< \tau$ then
            Train the network with stability and
              plasticity regularizers (Eq. 5)
        else
            $\theta^t$ = Expansion($\theta^t$)    ▷ Expand the
              network as described in
              Section 3.4 ;
            Train network with stability,
              plasticity and expansion
              regularizers (Eq. 6)
        end
    end
end
```

Continual Learning Scenario. Our experiments follow the standard continual learning scenario where a model is sequentially presented diverse tasks with unknown data distribution. We consider the situation where the model has access to D_{train}^t and D_{valid}^t only during the training period of task t. Afterwards, these data become unavailable, while D_{test}^t is used to evaluate the model's performance on $\mathcal{T}^t$ after learning $\mathcal{T}^{t'}$ with $t' > t$.

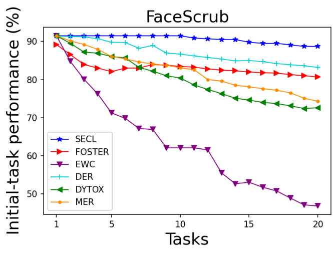
Benchmark Datasets. We conduct experiments on three image classification datasets that are popular continual learning benchmarks: CIFAR-10/100 (Krizhevsky et al., 2009), FaceScrub (Ng and Winkler, 2014), and ImageNet-50[1], which we generated as a subset of the ImageNet dataset. It contains 50 classes, which are grouped into two consecutive classes to form 25 tasks. Each class contains $1800/200$ 32×32 color training/test images.

For FaceSrub, which is a widely used dataset in face recognition, we select the first 100 people with the highest number of occurrences to form 100 classes, which we divide into 20 tasks with 5 classes each. In the case of CIFAR-10/100, the first task consists of all classes of CIFAR-10, while CIFAR-100 is

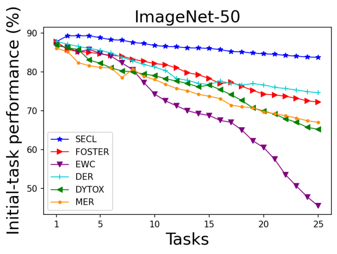
[1]Our ImageNet-50 dataset can be downloaded here.



(a) CIFAR-10/100.



(b) FaceSrub.



(c) ImageNet-50.

Figure 4: Assessment of catastrophic forgetting by measuring performance retention on the initial task. The results show, for each dataset, how the classification accuracy of the first task changes as subsequent tasks are learned. Overall, SECL shows the strongest resistance to catastrophic forgetting.

divided into 10 tasks, which serve as the remaining CL tasks, resulting in 11 tasks in this experiment.

Baselines. We compare our method to five state-of-the-art continual learning approaches including reference methods as well as recent, competitive ones: EWC (Kirkpatrick et al., 2017), MER (Riemer et al., 2018), DER (Yan et al., 2021), Dytox (Douillard et al., 2022), and FOSTER (Wang et al., 2022a). We perform grid search to fairly select the best hyperparameters for each approach.

Network. For FaceSrub and CIFAR-10/100, we use a CNN consisting of two blocks of 3×3 convolutions with $32, 64$ and 128 filters respectively (448 filters in total), followed by ReLU and 2×2 max-pooling. In the ImageNet-50 experiments, we use a CNN similar to (Vinyals et al., 2016), consisting of two blocks of 3×3 convolution with 64 filters, followed by ReLU and a 2×2 max-pooling. For all experiments we use a multi-headed network.

4.1 Is SECL Able to not Catastrophically Forget?

In a first experiment, we investigate the ability of our method to deal with the problem of catastrophic forgetting. A common measure of forgetting is to assess how the accuracy of a given task varies as the remaining tasks are learned (Mirzadeh et al., 2020). Here we choose the initial task and examine how its accuracy varies during the CL experiment. Since FaceSrub and ImageNet-50 contain a large number of tasks: 20 and 25 tasks respectively, i.e. 19 and 24 remaining tasks after the initial task, this setting is more relevant for both datasets. Nevertheless, we also show the results for CIFAR-10/100.

As can be seen in Figure 4, after sequential training on all tasks, SECL is the most stable and least forgetful on the ability to perform the initial task, showing little to no forgetting. We attribute this result to the adaptive stability regularizer (presented in Section 3.2), which prevents important parameters from being altered from task to task. DER (Yan et al., 2021) and FOSTER (Wang et al., 2022a) tends to be strong competitors to our method, especially on medium-sized datasets such as CIFAR-10/100, also archiving high levels of performance retention.

Interesting, SECL's retention capability slightly diminishes on FaceScrub and ImageNet compared to CIFAR-10/100. This underlines the fact that as the number of tasks in a continual learning experiment increases, it becomes more difficult to mitigate catastrophic forgetting. However, even with this minor setback in performance retention on these harder datasets, SECL consistently outperforms its competitors, demonstrating its robustness in overcoming catastrophic forgetting.

4.2 Is SECL a Competitive CL Approach in Terms of Accuracy?

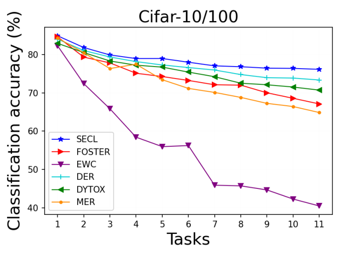
In a second experiment, we assess the effectiveness of our method in terms of classification accuracy during the continual learning experiments. To evaluate the classification accuracy, we use the all-important average accuracy metric (Lopez-Paz and Ranzato, 2017), which is common practice in the literature.

Specifically, the classification accuracy at $\mathcal{T}^t$, is the average accuracy obtained during testing on $\mathcal{D}^1_{test}, \cdots, \mathcal{D}^t_{test}$ and is defined as:

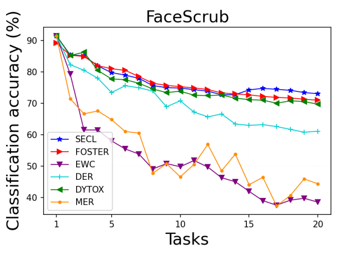
$$A^t = \frac{1}{t} \sum_{t'=1}^{t} a^{t,t'} \tag{7}$$

where $a^{t,t'}$ denotes the model's accuracy on $\mathcal{D}^{t'}_{test}$ after training on task t.

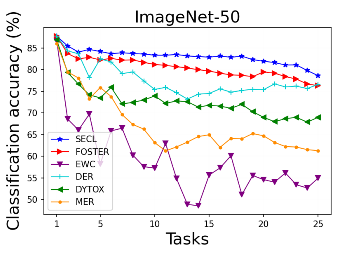
Figure 5 shows, for different datasets, how the average accuracy evolves across tasks during the continual learning experiments. The results indicate that SECL demonstrates strong performance in terms of



(a) CIFAR-10/100.

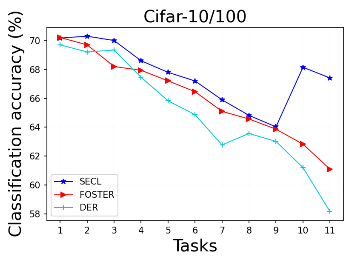


(b) FaceSrub.

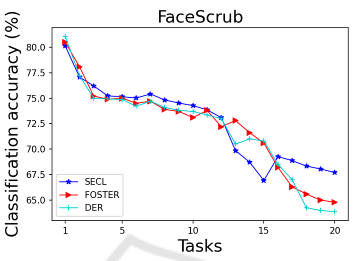


(c) ImageNet-50.

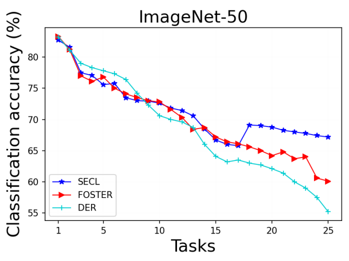
Figure 5: Evolution of the average test classification accuracy during the continual learning experiments on three datasets. SECL achieves significantly higher performance than the competing baselines.



(a) CIFAR-10/100.



(b) FaceSrub.



(c) ImageNet-50.

Figure 6: Evolution of the average accuracy on a much smaller network, which tends to reach its maximum capacity quickly. The effects of our network expansion strategy can be clearly seen in the later tasks in each dataset.

average accuracy compared to other baselines on all benchmarks. Interestingly, FOSTER (Wang et al., 2022a) achieves the best performance on the first 14 tasks on FaceSrub, before being clearly outperformed by our method on later tasks. This can be explained by the network expansion implemented in our approach, which gets triggered once the network's capacity nears its limit.

In-depth Analysis of Network Expansion Effects. In a third experiment, we closely examined the influence of the network expansion module on model accuracy. For this purpose, we reduce the size of the CNN to two 3×3 convolution blocks with 32 of filters each, followed by ReLU, a 2×2 max-pooling and a dense layer with 64 neurons. Given its limited size, we anticipated that the network would quickly reach its capacity limits, offering a transparent view of the network expansion's effects.

Figure 6 displays the evolution of the average classification accuracy for our approach alongside the FOSTER and DER baselines, both of which have demonstrated notable performance in prior experiments. As can be observed, the network capacity tends to saturate at the 9-th, 15-th and 17-th task for Cifar-10/100, FaceScrub and Imagenet-50 respectively. When these saturation points are reached, our expansion strategy comes into play, allowing the network to adapt and accommodate new tasks without overwhelming its capacity. At the same time, the network is further refined through our sparsification mechanism. This ensures that the newly added filters are used sparingly, optimizing their benefits and maintaining efficient performance across tasks.

An empirical conclusion that can be drawn from Figures 4, 5 and 6 is that SECL achieves strong overall continual learning results thanks to the way it addresses catastrophic forgetting and the learning of several new tasks through the stability-plasticity regularizers as well as the network expansion mechanisms built into the model learning procedure.

4.3 Ablation Study

We conducted an ablation study to investigate the contributions of each component in our SECL framework, particularly on the more challenging datasets, namely FaceScrub and ImageNet-50. In particular, we are interested in how (1) the stability regularizer governed by λ_s, (2) the plasticity regularizer governed by λ_p, and (3) the expansion regularizer governed by λ_e, contribute to the base model. We implement variants of SECL with different combinations of these components, each of which is either activated $(\surd)$ or removed $(\times)$. Furthermore, we also implement variants of SECL where two out of the three components were simultaneously deactivated to measure their combined impact on the model's performance.

Table 1: Average accuracy (A) and average forgetting (F) of SECL variants on FaceScrub and ImageNet-50 after learning the last task $\mathcal{T}^T$.

λ_s	λ_p	λ_e	FaceScrub		ImageNet-50	
			A(%)	F	A(%)	F
$\surd$	$\surd$	$\surd$	91.5	0.007	87.75	0.008
$\surd$	$\surd$	$\times$	88.2	0.081	75.1	0.175
$\times$	$\surd$	$\surd$	43.9	0.721	40.3	0.744
$\surd$	$\times$	$\surd$	68.93	0.044	64.7	0.078
$\times$	$\times$	$\surd$	36.3	0.728	34.8	0.742
$\surd$	$\times$	$\times$	39.2	0.12	37.6	0.194
$\times$	$\surd$	$\times$	38.2	0.731	37.93	0.771

The outcomes of this ablation study are summarized in Table 1, where we use two key metrics for evaluation: "A", which represents the average classification accuracy at $\mathcal{T}^T$ as defined in section 4.2, and "F", which denotes the average forgetting, defined as the average decrease in performance on each task between the peak accuracy and the accuracy after all tasks have been learned (Mirzadeh et al., 2020).

The results for these two metrics, shown in Table 1, demonstrate the effectiveness of each component not only in improving the average classification accuracy, but also in minimizing the level of forgetting across tasks. It is clear from these results that the interplay between the stability, plasticity, and expansion regularizers is critical to achieving the superior performance observed in our SECL framework.

5 CONCLUSION

In this work, we present SECL, a novel method for continual learning scenarios based on endowing a CNN's training objective with three key measures, respectively to stabilize the network on past tasks, to sparsify it to make it plastic for future tasks, and to extend it to learn more tasks once its capacity saturates. The experiments show that SECL outperforms baseline methods on standard continual learning image classification benchmarks, as well as on our custom imagenet benchmark of a longer continual learning scenario that spans 25 tasks.

REFERENCES

Aljundi, R., Babiloni, F., Elhoseiny, M., Rohrbach, M., and Tuytelaars, T. (2018). Memory aware synapses: Learning what (not) to forget. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 139–154.

Bonicelli, L., Boschini, M., Frascaroli, E., Porrello, A., Pennisi, M., Bellitto, G., Palazzo, S., Spampinato, C., and Calderara, S. (2023). On the effectiveness of equivariant regularization for robust online continual learning. *arXiv preprint arXiv:2305.03648*.

Caccia, L., Aljundi, R., Asadi, N., Tuytelaars, T., Pineau, J., and Belilovsky, E. (2022). New insights on reducing abrupt representation change in online continual learning. *arXiv preprint arXiv:2203.03798*.

Douillard, A., Cord, M., Ollion, C., Robert, T., and Valle, E. (2020). Podnet: Pooled outputs distillation for small-tasks incremental learning. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XX 16*, pages 86–102. Springer.

Douillard, A., Ramé, A., Couairon, G., and Cord, M. (2022). Dytox: Transformers for continual learning with dynamic token expansion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9285–9295.

Grossberg, S. (1987). Competitive learning: From interactive activation to adaptive resonance. *Cognitive science*, 11(1):23–63.

Gurbuz, M. B. and Dovrolis, C. (2022). Nispa: Neuro-inspired stability-plasticity adaptation for continual learning in sparse networks. *arXiv preprint arXiv:2206.09117*.

Hayes, T. L., Kafle, K., Shrestha, R., Acharya, M., and Kanan, C. (2020). Remind your neural network to prevent catastrophic forgetting. In *European Conference on Computer Vision*, pages 466–483. Springer.

Hering, A., Hansen, L., Mok, T. C., Chung, A. C., Siebert, H., Häger, S., Lange, A., Kuckertz, S., Heldmann, S., Shao, W., et al. (2022). Learn2reg: comprehensive multi-task medical image registration challenge, dataset and evaluation in the era of deep learning. *IEEE Transactions on Medical Imaging*, 42(3):697–712.

Hung, C.-Y., Tu, C.-H., Wu, C.-E., Chen, C.-H., Chan, Y.-M., and Chen, C.-S. (2019). Compacting, picking and growing for unforgetting continual learning. *Advances in Neural Information Processing Systems*, 32.

Kirkpatrick, J., Pascanu, R., Rabinowitz, N., Veness, J., Desjardins, G., Rusu, A. A., Milan, K., Quan, J., Ramalho, T., Grabska-Barwinska, A., et al. (2017). Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526.

Krizhevsky, A., Hinton, G., et al. (2009). Learning multiple layers of features from tiny images.

Lesort, T., Caselles-Dupré, H., Garcia-Ortiz, M., Stoian, A., and Filliat, D. (2019). Generative models from the perspective of continual learning. In *2019 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE.

Liu, H. and Liu, H. (2022). Continual learning with recursive gradient optimization. *arXiv preprint arXiv:2201.12522*.

Lopez-Paz, D. and Ranzato, M. (2017). Gradient episodic memory for continual learning. In *Advances in neural information processing systems*, pages 6467–6476.

Maschler, B., Pham, T. T. H., and Weyrich, M. (2021). Regularization-based continual learning for anomaly detection in discrete manufacturing. *Procedia CIRP*, 104:452–457.

Mirzadeh, S. I., Farajtabar, M., Pascanu, R., and Ghasemzadeh, H. (2020). Understanding the role of training regimes in continual learning. *Advances in Neural Information Processing Systems*, 33:7308–7320.

Ng, H.-W. and Winkler, S. (2014). A data-driven approach to cleaning large face datasets. In *2014 IEEE international conference on image processing (ICIP)*, pages 343–347. IEEE.

Ostapenko, O., Puscas, M., Klein, T., Jahnichen, P., and Nabi, M. (2019). Learning to remember: A synaptic plasticity driven framework for continual learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11321–11329.

Parikh, N. and Boyd, S. (2014). Proximal algorithms. *Foundations and Trends in optimization*, 1(3):127–239.

Parisi, G. I., Kemker, R., Part, J. L., Kanan, C., and Wermter, S. (2019). Continual lifelong learning with neural networks: A review. *Neural Networks*, 113:54–71.

Riemer, M., Cases, I., Ajemian, R., Liu, M., Rish, I., Tu, Y., and Tesauro, G. (2018). Learning to learn without forgetting by maximizing transfer and minimizing interference. *arXiv preprint arXiv:1810.11910*.

Rolnick, D., Ahuja, A., Schwarz, J., Lillicrap, T., and Wayne, G. (2019). Experience replay for continual learning. *Advances in Neural Information Processing Systems*, 32.

Rusu, A. A., Rabinowitz, N. C., Desjardins, G., Soyer, H., Kirkpatrick, J., Kavukcuoglu, K., Pascanu, R., and Hadsell, R. (2016). Progressive neural networks. *arXiv preprint arXiv:1606.04671*.

Sankararaman, K. A., De, S., Xu, Z., Huang, W. R., and Goldstein, T. (2020). The impact of neural network overparameterization on gradient confusion and stochastic gradient descent. In *International conference on machine learning*, pages 8469–8479. PMLR.

Schofield, D. P., Albery, G. F., Firth, J. A., Mielke, A., Hayashi, M., Matsuzawa, T., Biro, D., and Carvalho, S. (2023). Automated face recognition using deep neural networks produces robust primate social networks and sociality measures. *Methods in Ecology and Evolution*.

Schwarz, J., Czarnecki, W., Luketina, J., Grabska-Barwinska, A., Teh, Y. W., Pascanu, R., and Hadsell, R. (2018). Progress & compress: A scalable framework for continual learning. In *International Conference on Machine Learning*, pages 4528–4537. PMLR.

Serra, J., Suris, D., Miron, M., and Karatzoglou, A. (2018). Overcoming catastrophic forgetting with hard attention to the task. *arXiv preprint arXiv:1801.01423*.

Sharma, T., Debaque, B., Duclos, N., Chehri, A., Kinder, B., and Fortier, P. (2022). Deep learning-based object detection and scene perception under bad weather conditions. *Electronics*, 11(4):563.

Smith, J. S., Tian, J., Halbe, S., Hsu, Y.-C., and Kira, Z. (2023). A closer look at rehearsal-free continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2409–2419.

Sokar, G., Mocanu, D. C., and Pechenizkiy, M. (2021). Learning invariant representation for continual learning. *arXiv preprint arXiv:2101.06162*.

Vinyals, O., Blundell, C., Lillicrap, T., Wierstra, D., et al. (2016). Matching networks for one shot learning. *Advances in neural information processing systems*, 29:3630–3638.

Wang, F.-Y., Zhou, D.-W., Ye, H.-J., and Zhan, D.-C. (2022a). Foster: Feature boosting and compression for class-incremental learning. In *European conference on computer vision*, pages 398–414. Springer.

Wang, Z., Zhan, Z., Gong, Y., Yuan, G., Niu, W., Jian, T., Ren, B., Ioannidis, S., Wang, Y., and Dy, J. (2022b). Sparcl: Sparse continual learning on the edge. *Advances in Neural Information Processing Systems*, 35:20366–20380.

Yan, Q., Gong, D., Liu, Y., van den Hengel, A., and Shi, J. Q. (2022). Learning bayesian sparse networks with full experience replay for continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 109–118.

Yan, S., Xie, J., and He, X. (2021). Der: Dynamically expandable representation for class incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3014–3023.

Yoon, J., Madaan, D., Yang, E., and Hwang, S. J. (2021). Online coreset selection for rehearsal-based continual learning. *arXiv preprint arXiv:2106.01085*.

Yoon, J., Yang, E., Lee, J., and Hwang, S. J. (2017). Lifelong learning with dynamically expandable networks. *arXiv preprint arXiv:1708.01547*.

Yuan, M. and Lin, Y. (2006). Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 68(1):49–67.

Zeng, M., Xue, W., Liu, Q., and Guo, Y. (2023). Continual learning with dirichlet generative-based rehearsal. *arXiv preprint arXiv:2309.06917*.

Zhou, D.-W., Wang, Q.-W., Qi, Z.-H., Ye, H.-J., Zhan, D.-C., and Liu, Z. (2023). Deep class-incremental learning: A survey. *arXiv preprint arXiv:2302.03648*.

Zhou, G., Sohn, K., and Lee, H. (2012). Online incremental feature learning with denoising autoencoders. In *Artificial intelligence and statistics*, pages 1453–1461. PMLR.

Zhou, M., Xiao, J., Chang, Y., Fu, X., Liu, A., Pan, J., and Zha, Z.-J. (2021). Image de-raining via continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4907–4916.

Zhou, Y., Jin, R., and Hoi, S. C.-H. (2010). Exclusive lasso for multi-task feature selection. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 988–995. JMLR Workshop and Conference Proceedings.