SIESTA: Efficient Online Continual Learning with Sleep

Anonymous authors
Paper under double-blind review

Abstract

In supervised continual learning, a deep neural network (DNN) is updated with an ever-growing data stream. Unlike the offline setting where data is shuffled, we cannot make any distributional assumptions about the data stream. Ideally, only one pass through the dataset is needed for computational efficiency. However, existing methods are inadequate and make many assumptions that cannot be made for real-world applications, while simultaneously failing to improve computational efficiency. In this paper, we propose a novel online continual learning method, SIESTA based on wake/sleep framework for training, which is well aligned to the needs of on-device learning. The major goal of SIESTA is to advance compute efficient continual learning so that DNNs can be updated efficiently using far less time and energy. The principal innovations of SIESTA are: 1) rapid online updates using a rehearsal-free, backpropagation-free, and data-driven network update rule during its wake phase, and 2) expedited memory consolidation using a compute-restricted rehearsal policy during its sleep phase. For memory efficiency, SIESTA adapts latent rehearsal using memory indexing from REMIND. Compared to REMIND and prior arts, SIESTA is far more computationally efficient, enabling continual learning on ImageNet-1K in under 2.4 hours on a single GPU; moreover, in the augmentation-free setting it matches the performance of the offline learner, a milestone critical to driving adoption of continual learning in real-world applications.

1 Introduction

Training DNNs is incredibly resource intensive. This is true for both learning in highly resource constrained settings, e.g., on-device learning, and for training large production-level DNNs that can require weeks of expensive cloud compute. Moreover, for real-world applications, the amount of training data typically grows over time. This is often tackled by periodically re-training these production systems from scratch, which requires ever-growing computational resources as the dataset increases in size. Continual learning algorithms have the ability to learn from ever-growing data streams, and they have been argued as a potential solution for efficient learning for both embedded and large production-level DNN systems, improving the computational efficiency of network training and updating (Parisi et al., 2019). However, continual learning is rarely used for real-world applications because these algorithms fail to achieve comparable performance to offline retraining or they make assumptions that do not match real-world applications. As shown in Harun et al. (2023), many state-of-the-art continual learning methods e.g., BiC (Wu et al., 2019), WA (Zhao et al., 2020), and DER (Yan et al., 2021) are more expensive than offline models trained from scratch. Recent works have also argued that compute needs to be the focus of continual learning and that constraining memory serves little purpose except for on-device learning because storage costs are negligible compared to computation when training DNNs (Prabhu et al., 2023a,b; Hammoud et al., 2023). In this paper, we describe a resource efficient, continual learning algorithm that rivals an offline learner on supervised tasks, a critical milestone toward enabling the use of continual learning for real-world applications.

In conventional offline training of DNNs, training data is shuffled (making it independent and identically distributed (iid), which is required for stochastic gradient descent (SGD) optimization) and repeatedly looped through many training iterations. In contrast, an ideal continual learning algorithm is able to efficiently learn from *potentially* non-iid data streams, where each training sample is only seen by the learner once unless a limited amount of auxiliary storage is used to cache it. Most continual learning algorithms are designed solely to overcome catastrophic forgetting, which occurs when training with non-iid data (Parisi et al., 2019; Kemker et al., 2018). To do this, most models make implicit or explicit assumptions that go beyond the general supervised learning setting, e.g., some methods assume the

availability of additional information or assume a specific structure of the data stream. Moreover, existing methods do not match the performance of an offline learner, which is essential for industry to adopt continual learning for updating large DNNs.

We argue that a continual learning algorithm should have the following properties:

1. It should be capable of online learning and inference in a compute and memory constrained environment,
2. It should rival (or exceed) an offline learner, regardless of the structure of the training data stream,
3. It should be significantly more computationally efficient than training from scratch, and
4. It should make no additional assumptions that constrain the supervised learning task, e.g., using task labels during inference.

These criteria are simple; however, most continual learning algorithms make strong assumptions that do not match real-world systems and are assessed on toy problems that are not appropriate surrogates for real-world problems where continual learning could greatly improve computational efficiency. For example, many works still focus on tasks such as permuted MNIST and split-CIFAR100 (Chaudhry et al., 2018b; 2019, Rahaf & Lucas, 2019; Pan et al., 2020, Titsias et al., 2019; Zenke et al., 2017; Rajasegaran et al., 2019), only work in extreme edge cases like incremental class learning (Castro et al., 2018; Chaudhry et al., 2018b, Hou et al., 2019; Rebuffi et al., 2017; Tao et al., 2020; Wu et al., 2019), assume the availability of task-labels during inference (Golkar et al., 2019, Fernando et al., 2017, Hung et al., 2019; Serra et al., 2018), or require large batches to learn (Yan et al., 2021; Douillard et al., 2022). For continual learning to have practical utility, we need efficiency and performance that rivals trained from scratch models, as well as robustness to data ordering.

There are two extreme frameworks for continual learning. At one extreme, is incremental batch learning where the agent receives a batch and has as much time as necessary to loop over that batch before proceeding to the next batch. Typically these systems are evaluated with large batches, and many experience dramatic performance decreases when smaller batches are used (Hayes et al., 2020). This setting is often studied in class incremental learning and domain incremental learning. At the other extreme is online learning, where the agent receives one input at a time that must be immediately learned. Humans and animals learn in a manner that is a compromise between these two extremes. They acquire new experiences in an online manner and these experiences are consolidated offline during sleep (McClelland & Goddard, 1996; Hayes et al., 2021). Sleep plays a role in memory consolidation in all animals studied, including invertebrates, birds, and mammals (Vorster & Born, 2015). While animals sleep to consolidate memories, they can use both consolidated (post-sleep) and recent (pre-sleep) experiences to make inferences while awake.

Although this paradigm is virtually ubiquitous among animals, it has rarely been studied as a paradigm in continual learning. This paradigm matches a real-world need: on-device continual learning and inference (Hayes & Kanan, 2022). For example, virtual/augmented reality (VR/AR) headsets could use continual learning to establish play boundaries and to identify locations in the physical world to augment with virtual overlays. Home robots, smart appliances, and smart phones need to learn about the environments and the preferences of their owners. In all of these examples, learning online is needed, but there are large periods of time where offline memory consolidation is possible, e.g., while the mobile device is being charged or its owner is asleep. In this paper, we formalize this paradigm for continual learning and we describe an algorithm with these capabilities, which we call SIESTA (**S**leep **I**ntegration for **E**pisodic **ST**re**A**ming).

For memory and computational efficiency, SIESTA adopts the quantized latent rehearsal scheme from REMIND (Hayes et al., 2020). REMIND continually trains the upper layers of a deep neural network (DNN) in a pseudo-online manner. It stores quantized mid-level representations of seen inputs in a buffer, which enables it to store a much larger number of samples with a given memory budget compared to veridical rehearsal methods that store raw images. To do pseudo-online training, REMIND uses rehearsal (Hetherington, 1989), a method for mitigating catastrophic forgetting by mixing new inputs with old inputs. For every new input, REMIND reconstructs a small number of past inputs, mixes the new input with them, and updates the DNN with this mini-batch; however, using rehearsal for every sample to be learned is not ideal. SIESTA addresses this by using rehearsal only during its offline sleep stage. For online learning, SIESTA instead uses lightweight online updates of the DNN's output layer.

Our major contributions are summarized as:

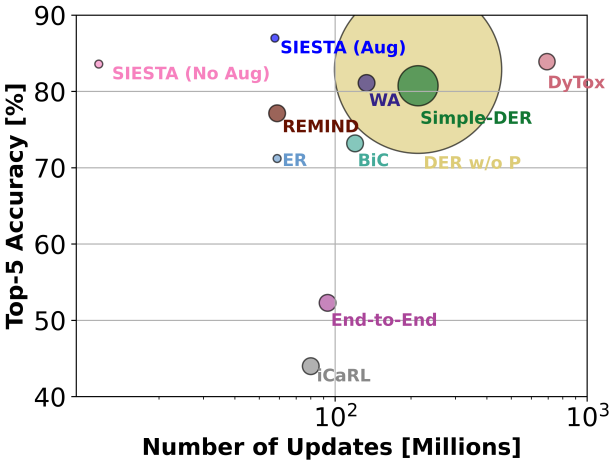


Figure 1: Our method, SIESTA, outperforms existing continual learning methods for class-incremental learning on ImageNet-1K while requiring fewer network updates and using fewer parameters, as denoted by circle size.

1. We formalize a framework for online updates with offline memory consolidation, and we describe the SIESTA algorithm that operates in this framework (see Figure 3). SIESTA is capable of rapid online learning and inference while awake, but has periods of sleep where it performs offline memory consolidation.

2. For incremental class learning on ImageNet-1K, SIESTA achieves state-of-the-art performance using far fewer parameters, memory, and computational resources than other methods. Without augmentations, training SIESTA requires only 2.4 hours on a single NVIDIA A5000 GPU. In contrast, recent methods require orders of magnitude more compute (see Figure 1).

3. SIESTA is the first continual learning algorithm to achieve identical performance to an offline model, when augmentations are not used. It solves *catastrophic forgetting* - the most studied problem in continual learning (see Table 1). SIESTA is capable of working with arbitrary orderings, and achieves similar performance in both class incremental and iid settings.

2 Online Updates with Offline Consolidation

We formalize the classification problem setting for supervised online continual learning with offline consolidation. The learner alternates between an online phase (wake) and an offline phase (sleep). For learning, during the j'th online phase, the agent receives a sequence of n labeled observations, i.e., $t_{j1}, t_{j2}, \ldots, t_{jn}$, where each input observation x_t has label y_t. The sequence is not assumed to be stationary, and it can contain examples from classes from an arbitrary label ordering. The agent can cache these labeled pairs, or a subset of them, in memory with storage of size b bits. The agent can be evaluated at any time during the online phase, where it must make inferences using both recent experiences from the j'th online phase, as well as past experiences from previous phases. During the subsequent offline consolidation phase, the agent is allowed at most m updates of the network, e.g., gradient descent updates. We do not assume task labels are available during any phase. In general, iid (shuffled) orderings do not cause catastrophic forgetting; and at the other extreme, an ordering sorted by category causes severe catastrophic forgetting in conventional algorithms (Kemker et al., 2018).

With some exceptions (Kemker & Kanan, 2018; Pham et al., 2021; 2023; Arani et al., 2022), this paradigm has been little studied and offers several advantages. For example, it allows embedded mobile devices to quickly use new information from users and their environments and then consolidate that learning during a scheduled downtime. It also serves as a setting for studying learning efficiency in continual learning, where different sleep policies can be studied for minimizing the number of updates m during a sleep setting. Lastly, it allows for testing functional hypotheses from neuroscience about sleep. While rehearsal-like mechanisms used in continual learning occur during slow wave sleep, the mechanisms that occur during rapid eye movement (REM) sleep have not yet been explored with DNNs nor

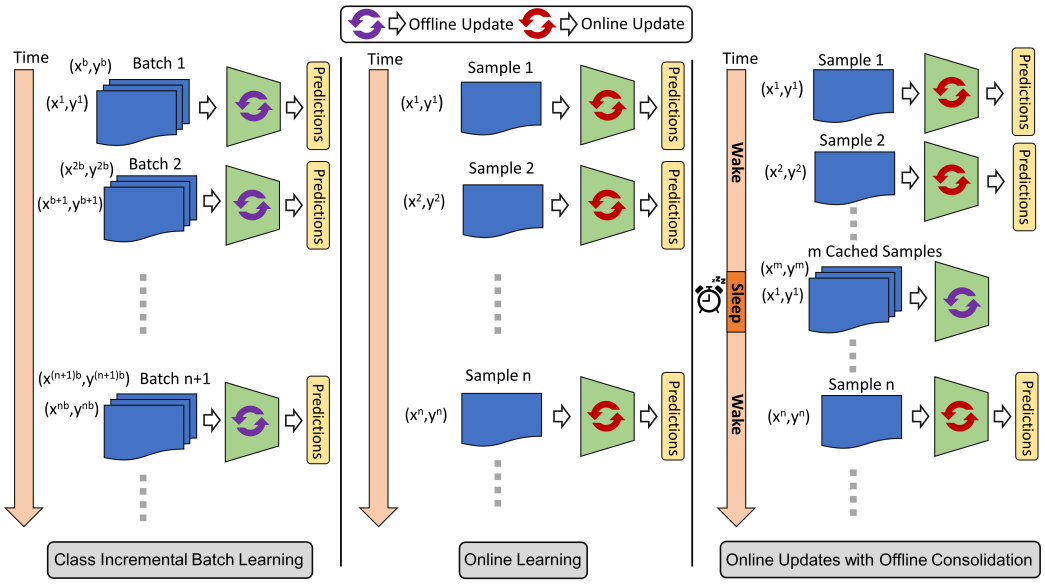


Figure 2: An overview of online updates with offline consolidation paradigm. While awake, agent performs online learning and while asleep, it performs computationally restricted offline learning. This wake/sleep cycles oscillate. Thus, our paradigm can be viewed as a combination of class incremental batch learning and online learning paradigms.

has the interplay between slow wave sleep and REM sleep. REM sleep increases abstraction, facilitates pruning of synapses, and is when dreams occur (Smith & Smith, 2003; Djonlagic et al., 2009; Cai et al., 2009; Lewis et al., 2018; Durrant et al., 2015; Li et al., 2017). We juxtapose our training paradigm with alternatives in continual learning in Section 3.

Unlike incremental batch learning, in SIESTA compute is restricted during the sleep cycle. This enables us to explicitly model and control the amount of compute used during each rehearsal cycle. Traditional rehearsal is similar to our concept of sleep, where the model pauses after observing a certain number of samples to mix in new samples. An overview of this paradigm is juxtaposed with existing paradigms in Figure 2.

3 Related Work

We compare SIESTA's online updates with offline consolidation paradigm to alternative paradigms.

Task Incremental Learning with Task Labels. Incremental task batch learners (Kirkpatrick et al., 2017; Zenke et al., 2017; Aljundi et al., 2018; Chaudhry et al., 2018a;b; Serra et al., 2018; Dhar et al., 2019), learn from task batches, where each batch has a distinct task that is often a binary classification problem. These methods assume the task label is available during evaluation so that the correct "output" head can be selected, and when this assumption is violated these methods fail (Hayes & Kanan, 2020; Hayes et al., 2020). SIESTA does not require task labels for prediction, which are typically not available in real-world applications.

Class Incremental Batch Learning. In this paradigm, a dataset is split into multiple batches, where each batch consists of mutually exclusive categories, without any revisiting of categories. An agent is given a batch to learn for as long as it likes (see Figure 2) and typically can use some auxiliary memory for rehearsal. This paradigm has been studied with rehearsal-based methods (Hayes et al., 2021; Abraham & Robins, 2005; Belouadah & Popescu, 2019; Castro et al., 2018; Chaudhry et al., 2018b; French, 1997; Hayes et al., 2019; 2020; Hou et al., 2019; Rebuffi et al., 2017; Tao et al., 2020; Wu et al., 2019) that store previously observed data in a memory buffer or reconstruct them to rehearse alongside new data. It has also been studied in regularization based methods (Chaudhry et al., 2018b; Aljundi et al., 2018; Chaudhry et al., 2018a; Dhar et al., 2019; Kirkpatrick et al., 2017; Fernando et al., 2017; Coop et al., 2013; Li & Hoiem, 2017; Lopez-Paz & Ranzato, 2017; Ritter et al., 2018; Serra et al., 2018; Zenke et al., 2017) that

constrain new weight updates to penalize large deviations from past weights, as well as dynamic methods (Douillard et al., 2022; Draelos et al., 2017; Yoon et al., 2018; Hou et al., 2018; Ostapenko et al., 2019; Rusu et al., 2016; Yan et al., 2021) that incrementally increase the capacity of a DNN over time. While task labels are not used during prediction, evaluation takes place between batches and many methods require large batches (e.g., thousands of examples) or they fail (Hayes et al., 2020). Some methods use a large number of parameters for keeping copies of the network in memory for distillation (Castro et al., 2018; Kang et al., 2022). We argue that while class incremental learning is a valuable assessment of a continual learner's ability to avoid catastrophic forgetting given an extremely adversarial data ordering, there is little real-world utility in algorithms that are designed solely to do class incremental learning. SIESTA differs from algorithms designed for this paradigm in that it can perform inference at any time and can operate for arbitrary data orderings, including when classes are revisited.

Online Learning. Unlike batch learning paradigms, in online learning, an agent observes data sequentially and learns them instance-by-instance in a single pass through the dataset (see Figure 2). To study catastrophic forgetting, examples are typically ordered by class, although alternative orders are sometimes studied (Hayes et al., 2020). Evaluation can occur at any point during training. This setting eliminates looping over data many times and evaluation between batches, thus making it more memory and compute time efficient, which is desirable for embedded devices. This paradigm has primarily been studied on smaller datasets (CIFAR-100) (Lopez-Paz & Ranzato, 2017; Chaudhry et al., 2018b; Rahaf & Lucas, 2019; Wang et al., 2021), although some methods have been shown to scale to ImageNet-1K (Hayes & Kanan, 2020; Hayes et al., 2019; Hayes & Kanan, 2020; Gallardo et al., 2021). For ImageNet-1K, these methods under-perform incremental batch learning methods (Hayes et al., 2020). SIESTA's paradigm is a compromise that captures most of the benefits of online continual learning while enabling increased accuracy.

Paradigm Relationships. The formal online updates with offline consolidation setting can be configured to mirror other continual learning settings (see Figure 2). For class incremental batch learning, where the learner receives large batches of training examples (e.g., $n > 100,000$ for ImageNet-1K (Rebuffi et al., 2017; Yan et al., 2021)), buffer $b = b_{\text{recent}} + b_{\text{buffer}} + b_{\text{auxiliary}}$ would be sufficiently large to hold all n observations from the j'th online phase (b_{recent}) as well as past examples used for rehearsal (b_{buffer}), and any additional memory needed for other purposes ($b_{\text{auxiliary}}$) (e.g., distillation), and m is typically very large (e.g., for the state-of-the-art method DyTox (Douillard et al., 2022), $m > 500n$). A pseudo-online algorithm like REMIND (Hayes et al., 2020) uses a configuration of $n = 1$ and $m = 51$. For both REMIND and SIESTA, b only acts as a buffer for storing compressed past observations and their labels.

4 The SIESTA Algorithm

The SIESTA algorithm (Figure 3) alternates between awake and sleep phases. The awake phase involves online learning as well as sample compression, storage, and inference. The sleep phase involves memory consolidation via brief periods of offline learning. SIESTA is designed to handle data streams with arbitrary class orders, ranging from iid to class incremental paradigms.

SIESTA is a feed-forward DNN defined as $\mathcal{F}(\mathcal{G}(\mathcal{H}(\cdot)))$, where $\mathcal{H}(\cdot)$ contains the bottom layers, $\mathcal{G}(\cdot)$ contains the top layers prior to the output layer, and $\mathcal{F}(\cdot)$ is the output layer. Specifically, SIESTA takes as input a 3rd-order tensor $\mathbf{X}$. $\mathcal{H}(\cdot)$ produces $\mathbf{Z} = \mathcal{H}(\mathbf{X})$, where $\mathbf{Z} \in \mathbb{R}^{r \times s \times d}$, r and s are the tensor spatial dimensions, and d are the tensor channel dimensions. This tensor is then transformed into a vector embedding, i.e., $\mathbf{z} = \mathcal{G}(\mathbf{Z})$. The output layer $\mathcal{F}(\cdot)$ is then described with cosine softmax, where the score for the k'th class is given by:

$$p_k = \frac{\exp\left(a_k \tau^{-1}\right)}{\sum_j \exp\left(a_j \tau^{-1}\right)}, \tag{1}$$

where

$$a_k = \frac{\mathbf{f}_k^T \mathbf{z}}{\|\mathbf{f}_k\|_2 \|\mathbf{z}\|_2}, \tag{2}$$

$\mathbf{f}_k$ is the weight vector for the k'th class, and $\tau \in \mathbb{R}$ is a learned temperature used during optimization. It has been proven that cosine softmax encourages greater class separation than softmax (Kornblith et al., 2021). In our implementation, $\mathcal{H}$ and $\mathcal{G}$ are both convolutional networks; however, other architectural choices would be suitable.

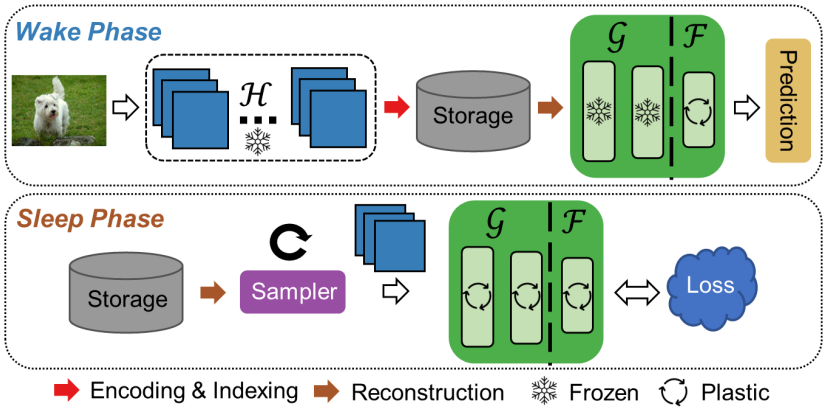


Figure 3: A high-level overview of SIESTA. During the *Wake Phase*, it transforms raw inputs into intermediate feature representations using network $\mathcal{H}$. The inputs are then compressed with tensor quantization and cached. Then, weights belonging to recently seen classes in network $\mathcal{F}$ are updated with a running class mean using the output vectors from $\mathcal{G}$. Finally, inference is performed on the current sample. During the *Sleep Phase*, a sampler uses a rehearsal policy to choose which examples should be reconstructed from the cached data for each mini-batch. Then, networks $\mathcal{G}$ and $\mathcal{F}$ are updated with backpropagation in a supervised manner. The wake/sleep cycles alternate.

Following Hayes et al. (2020), prior to continual learning, the DNN is initialized by pre-training on an initial set of N training samples, e.g., images from the first 100 classes of ImageNet. Tensor features $\mathbf{Z}$ are extracted from each of the N samples and used to fit a Product Quantization (PQ) model (Jegou et al., 2010) to all rsN d-dimensional vectors in these tensors. This enables us to efficiently store and reconstruct compressed representations of $\mathbf{Z}$. This approach enables SIESTA to much more efficiently use memory for rehearsal than methods that store raw images. The network $\mathcal{H}(\cdot)$ is then kept fixed during continual learning. While this aspect of SIESTA is the same as REMIND, SIESTA differs significantly in its capabilities and how it is trained. REMIND uses rehearsal during online learning by sampling a minibatch that has 50 old examples and the currently observed example. Instead, SIESTA does not use rehearsal for its online updates and it only employs rehearsal during its sleep phase. We next describe how SIESTA's two learning phases operate.

4.1 Online Learning while Awake

During the awake phase (see Figure 3), only the output layer $\mathcal{F}$ of the DNN is updated. This enables SIESTA to avoid catastrophic forgetting and permits lightweight online updates. When SIESTA receives an input tensor $\mathbf{X}_t$ at time t, $\mathbf{Z}_t$ is then compressed and saved in a limited-sized storage buffer using PQ along with its class label. If the buffer is full, then a randomly selected sample is removed from the class with the most samples. Subsequently, the output layer weights are updated with simple running updates for the appropriate class. The update for the output layer weight vector for class k is given by

$$\mathbf{f}_k \leftarrow \frac{c_k \mathbf{f}_k + \mathbf{z}_t}{c_k + 1}, \tag{3}$$

where c_k is an integer counter for class k. After updating the weight vector, c_k is incremented, i.e., $c_k \leftarrow c_k + 1$. For inference, the class with the highest score p_k in Equation 1 is selected as the predicted class.

4.2 Memory Consolidation During Sleep

During the sleep phase (see Figure 3), the output layer $\mathcal{F}$ and the top layers $\mathcal{G}$ are trained using rehearsal, while the bottom layers $\mathcal{H}$ are kept frozen. Rehearsal consists of selecting mini-batches of stored examples in the buffer for reconstruction and then using them to update the network with backpropagation. Following the paradigm in Section 2, the DNN is allowed at most m gradient descent updates. Given a mini-batch of size q, each sleep cycle therefore consists of updating the DNN with n mini-batches, where the total number of updates is $m = q \times n$. At the beginning of a sleep cycle, the samples chosen for reconstruction are governed by a policy. Our main results all use balanced

uniform sampling, i.e., sampling an equal number from each class, which worked best on class balanced datasets and was competitive on long-tailed ones. Other policies are studied in Appendix D.

In a subset of our experiments, we use augmentation during learning. While augmentation is typically applied directly to images, here we apply it to the reconstructed $\mathbf{Z}$ tensors. We use two forms of augmentation: manifold mix-up (Verma et al., 2019) and cut-mix (Yun et al., 2019). Both strategies are used in the standard manner, except instead of producing a weighted combination of two images, we create a weighted combination of tensors.

In our main results, we have the network sleep every $120K$ samples, which in the incremental class learning setting corresponds to training on 100 categories. We study the impact of sleep frequency and sleep length in Section 6.3.

4.3 Network Architecture & Initialization

While continual learning is starting to use transformers (Douillard et al., 2022), recent work has primarily used ResNet18 (Rebuffi et al., 2017; Wu et al., 2019; Castro et al., 2018; Wu et al., 2019; Hayes et al., 2020; Yan et al., 2021). However, ResNet18 has been shown to perform worse than other similarly sized DNNs (Hayes & Kanan, 2022). Moreover, given one of the major applications of continual learning is on-device learning, using a DNN designed for embedded devices is ideal. Therefore, in our main results, we use MobileNetV3-L (Howard et al., 2019). MobileNetV3-L (5.48M) is lightweight with $2\times$ fewer parameters than ResNet18 (11.69M) and has lower latency. Since PQ encodes features across channels, MobileNetV3-L is more suitable for compressing its features with relatively less reconstruction error than ResNet18. We compare MobileNetV3-L and ResNet18 in Appendix E.

In Gallardo et al. (2021), pre-training (base initialization) with the self-supervised learning (SSL) algorithm SwAV (Caron et al., 2020) outperformed supervised pre-training for continual learning. We use their pre-training method in our main results, but we study other SSL methods in Appendix K. For fair comparisons, we use the same SwAV pre-trained DNN with DER (Yan et al., 2021), ER (Chaudhry et al., 2019), and REMIND (Hayes et al., 2020). Using MobileNetV3-L, we set network $\mathcal{H}$ to be the first 8 layers of the network, consisting of 2.19% of the network parameters, which was found to be the best balance of accuracy and efficiency (see Appendix F). Using images of size 224×224 pixels, $\mathcal{H}$ produces a tensor $\mathbf{Z} \in \mathbb{R}^{14\times14\times80}$. For PQ, we use *Optimized Product Quantization (OPQ)* from FAISS (Johnson et al., 2019), which is used to compress and reconstruct the 14^2 80-dimensional vectors that make up each tensor. Following REMIND, we exclusively use reconstructed versions of the output of $\mathcal{H}$ during continual learning. The remaining 11 layers of MobileNetV3-L (97.81% of the DNN parameters) are trained during sleep.

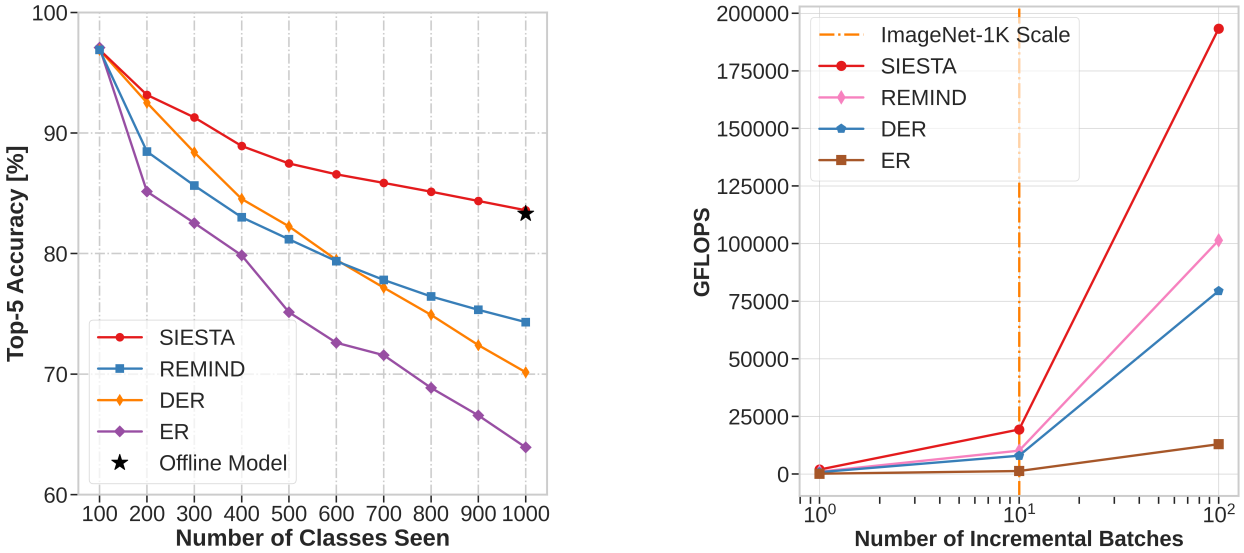
5 Experimental Setup

Comparison Models. We compare SIESTA to a variety of baseline and state-of-the-art methods, including online learners REMIND (Hayes et al., 2020), ER (Chaudhry et al., 2019), SLDA (Hayes & Kanan, 2020), NCM (Mensink et al., 2013); incremental batch learners iCaRL (Rebuffi et al., 2017), BiC (Wu et al., 2019), End-to-End (Castro et al., 2018), WA (Zhao et al., 2020), Simple-DER (Li et al., 2021), DER (Yan et al., 2021), DyTox (Douillard et al., 2022); and an offline learner. We compare with two variants of DER: DER without pruning (referred to as DER†) and DER with pruning (referred to as DER$*$). These methods have been designed to be effective for incremental class learning on ImageNet-1K, and more details are in Appendix B. SIESTA is trained with cross-entropy loss and uses SGD as its optimizer with the OneCycle learning rate (LR) scheduler (Smith & Topin, 2019) during each sleep phase. It uses a higher initial LR in the last layer to help learn the new tasks and a lower LR in earlier layers to mitigate forgetting of previously learned information. For each sleep cycle, we use a batch size of 64, momentum 0.9, weight decay $1e$-5, and an initial LR of 0.2 for the last layer. LR is reduced in earlier layers by a layer-wise decay factor of 0.99. Additional details of SIESTA and implementation details for other methods are provided in Appendix C.

Datasets and Evaluation Criteria. For our main results, we use ImageNet ILSVRC-2012 (Russakovsky et al., 2015), which is the standard object recognition benchmark for testing a model's ability to scale. It has 1.28 million images uniformly distributed across 1000 categories. We also use a long-tailed version of Places-2 Dataset (Zhou et al., 2017) to evaluate rehearsal policies (see Appendix D for details). We configure the datasets in the continual iid setting and the class incremental learning setting, where data is ordered by class and images are shuffled within each class. For evaluation, we report the average accuracy μ over all steps T, where $\mu = \frac{1}{T}\sum_{t=1}^{T} \alpha_t$, with α_t referring to the accuracy at step t. We also report the final accuracy α for continual learning models. Note that α means best

Table 1: Class incremental learning results on ImageNet-1K. For a fair comparison, we constrain methods to 12.5 million updates and do not use data augmentation. The (↑) and (↓) indicate high and low values to reflect optimum performance respectively. $\mathcal{P}$ is the number of parameters in Millions, μ is the average top-5 accuracy, α is the final top-5 accuracy, $\mathcal{M}$ is the total memory in GB, and $\mathcal{U}$ is the total number of updates in Millions.

Method	$\mathcal{P}(\downarrow)$	$\mu(\uparrow)$	$\alpha(\uparrow)$	$\mathcal{M}(\downarrow)$	$\mathcal{U}(\downarrow)$	GFLOPS (↑)
Offline	5.48	—	83.31	192.87	768.70	—
DER	54.80	81.87	70.15	20.99	12.43	7944.60
ER	5.48	76.32	63.92	19.59	11.53	1294.10
REMIND	5.48	81.77	74.31	2.02	11.53	10139.00
SIESTA	**5.48**	**88.33**	**83.59**	**2.02**	**11.53**	**19326.00**



(a) ImageNet-1K learning curve

(b) GFLOPS

Figure 4: Comparison among SIESTA and baselines without augmentations. **(a)** We show learning curves on ImageNet-1K comparing continual learners with an offline learner. **(b)** We also compare continual learners based on GFLOPS. Each incremental batch corresponds to 100 classes from ImageNet-1K.

accuracy for offline models. We use top-5 accuracy for ImageNet-1K and top-1 accuracy for Places. Additionally, we measure the total number of parameters $\mathcal{P}$ (in millions) and the total memory $\mathcal{M}$ (in gigabytes) used by each model. We also measure the total number of updates $\mathcal{U}$, where an update consists of a backward pass on a single input.

6 Results

6.1 Comparison with the Offline Learner

We first compare SIESTA, ER, DER, and REMIND to an offline learner. For many real-world applications, a continual learner that performs significantly worse than an offline learner is unacceptable. Moreover, for these applications we cannot make assumptions about the class order distribution of the data stream; so for SIESTA, ER, and REMIND, we study both the continual iid and class incremental learning settings, which can be seen as extreme best case and extreme worst case scenarios respectively. DER, as designed, is only capable of class incremental learning. All compared methods use the same SwAV pre-trained MobileNetV3-L DNN on the first 100 classes of ImageNet-1K. The offline learner is a MobileNetV3-L trained from scratch.

We first show results where all models omit augmentations, including the offline learner. SIESTA used 1.28M updates per sleep cycle. All continual learners used a similar total number of updates. Results for class incremental learning

are given in Table 1 and learning curves in Figure 4a. DER, ER, and REMIND perform over 9% worse (absolute) than the offline learner for final accuracy. In contrast, SIESTA matches the offline learner's performance for final accuracy.

We analyzed SIESTA, ER, and REMIND under the continual iid setting and compare it against the class incremental setting. For the iid setting, SIESTA obtains 83.45% in final accuracy, whereas ER and REMIND attain 64.92% and 79.52% in final accuracy respectively. When switching from iid to the class incremental setting ER and REMIND decrease 1% (absolute) and 5.21% (absolute) in final accuracy respectively. In contrast, SIESTA maintains similar performance in both settings and shows robustness to data ordering. To further analyze SIESTA's performance across these distributions compared to an offline model, we used Chochran's Q test (Conover, 1999), a non-parametric paired statistical test that can be used for comparing three or more classifiers, and we found no significant difference among SIESTA's final accuracy for the iid and class incremental settings compared to the offline learner ($P = 0.08$). Therefore, SIESTA achieves *"no forgetting"* by matching the performance of the offline MobileNetV3-L across orderings.

In augmentation experiments, SIESTA outperforms DER, ER, and REMIND by 15.18%, 15.78%, and 4.03% (absolute) respectively in final accuracy (see Table 6). We also compare SIESTA with its awake-only variant and other online learning methods including REMIND, ER, SLDA, and NCM in Appendix H where SIESTA outperforms all compared methods and SIESTA (awake-only) shows competitive performance. SIESTA was found to be robust to data ordering in the augmentation setting as well. Using a McNemar's test on the final accuracy for SIESTA in the iid and class incremental settings revealed no significant difference ($P = 0.85$). However, with augmentations there was a significant difference between the offline learner and SIESTA based on a McNemar's test ($P < 0.001$). Additional details are in Appendix G. We hypothesize the offline model outperformed SIESTA due to the features learned in network $\mathcal{H}$ being not sufficiently universal (see Appendix K).

6.2 Computational & Memory Efficiency

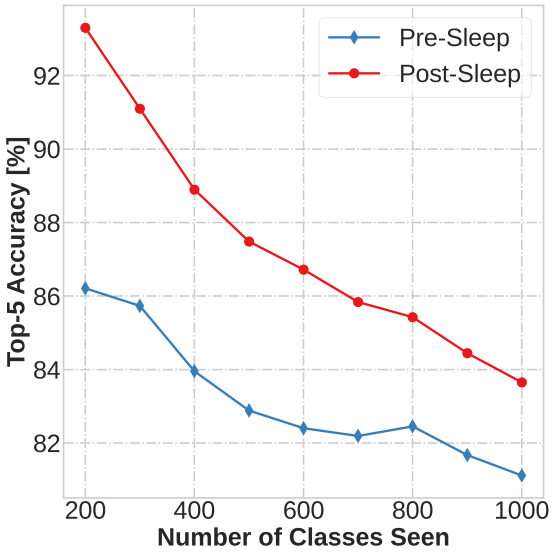
A major goal of SIESTA is efficiency via continual learning, whereas the vast majority of continual learning systems are not more efficient than retraining an offline model (Harun et al., 2023). To assess computational efficiency of models, we exclude the pre-training phase, since it only occurs once and our focus is enabling continual learning to be significantly more efficient than periodic retraining from scratch.

In terms of memory efficiency, REMIND and SIESTA use memory to store compressed tensors and the others use it to store 20,000 images. As shown in Table 1 and Table 2, SIESTA requires $10\times$ less memory than other methods ($19 - 22$ gigabytes) except REMIND. Moreover, SIESTA requires $2 \times -20\times$ fewer parameters than most methods ($11.68 - 116.89$ million) (see Table 2). SIESTA is much more suitable for memory constrained real-world applications especially on-device learning than most other methods.

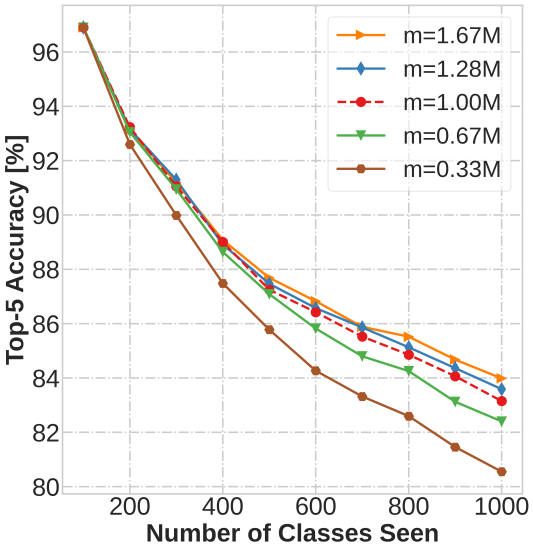
Based on the number of total DNN updates, SIESTA is much more computationally efficient than other methods. It requires $7 \times -60\times$ fewer DNN updates than others (see Table 2). We also empirically compared REMIND and SIESTA's continual learning training time when learning 900 ImageNet-1K classes using a single NVIDIA RTX A5000 GPU. Augmentations were not used with either model. REMIND required 8.1 hours. Using a batch size of 64 during sleep, which we used in the experiments in Section 6.1, SIESTA requires only 2.4 hours ($3.4\times$ faster than REMIND). When the batch size is increased to 512, SIESTA required only 1.9 hours with a negligible change in accuracy (see Appendix J).

Using DeepSpeed[1] with the same GPU across models, we also conducted a computational analysis based on FLOPS (floating-point operations per second) which revealed that SIESTA has $1.9\times$, $2.4\times$, and $14.9\times$ higher GFLOPS than REMIND, DER, and ER respectively (see Table 1). Unlike other methods, SIESTA does not train the DNN during online learning and only performs a fixed number of backpropagation updates per sleep ($m = 1.28M$), thereby providing much higher GFLOPS than compared baselines. In real-world continual learning, we could potentially have an infinitely long (never ending) data stream, which would be larger than ImageNet-1K, raising the question: *how well do these models computationally scale to larger datasets?* In Figure 4b, we extrapolate from our FLOPS analysis to even larger datasets, which demonstrates that the gap in GFLOPS between SIESTA and other methods grows significantly, where SIESTA is predicted to be much more efficient than others in the large-scale dataset regime.

[1] https://github.com/microsoft/DeepSpeed



(a) Pre vs. post sleep study



(b) Sleep length study

Figure 5: Analysis of sleep in SIESTA (No Aug). **(a)** We study the overall impact of sleep using our default model configuration by showing the learning curves of pre- and post-sleep performances as a function of seen classes. **(b)** We also study the impact of sleep length m on SIESTA's performance.

6.3 Sleep Analyses

We asked the question *"What is the impact of sleep on SIESTA's ability to learn and remember?"* These experiments are for incremental class learning, and data augmentation is not used. First, we analyze the pre-sleep and post-sleep performance of SIESTA on ImageNet-1K in Figure 5a using the same sleep settings as in Section 6.1. We see that the performance of SIESTA after sleep is consistently higher than before sleep for all increments, providing a $4.25 \pm 1.38\%$ average absolute increase in accuracy after each sleep cycle.

Next, we study the impact of sleep frequency. To do this, we trained models with a sleep frequency of 50, 100 (default), or 150 classes seen. For this analysis, we used a fixed sleep length, i.e., a fixed number of updates per sleep ($m = 1.28M$). The absolute average increase in post-sleep accuracy was $2.29 \pm 0.86\%$ for 50 classes, $4.25 \pm 1.38\%$ for 100 classes, and $6.18 \pm 2.15\%$ for 150 classes. Despite the 50 class increment being trained with more updates, the 100 class increment achieved better post-sleep performance. We hypothesize that this happens because frequent sleep leads to greater perturbation in the DNN's weights, resulting in gradual forgetting of old memories.

In Figure 5b, we study the impact of sleep length on SIESTA's performance by varying the number of updates (m) during each sleep period, where SIESTA slept every 100 classes. We observe that as sleep length increases, SIESTA's performance also increases; however, as the sleep length increases, SIESTA requires more updates and there are diminishing returns in terms of increases in accuracy, so we must strike a balance between accuracy and efficiency.

6.4 State-of-the-Art Comparisons

To put our work in context with respect to existing methods, we compare SIESTA against recent class incremental learning methods that have previously been shown to perform well on ImageNet-1K. All methods, except for SIESTA (No Aug), use augmentations and have a variety of different DNN architectures. SIESTA (Aug), which uses augmentations, used 6.4 million updates per sleep cycle ($m = 6.4M$). With the exception of REMIND, ER, and SIESTA, we use published performance numbers for all methods. REMIND and SIESTA both use around 2GB of memory for rehearsal and SwAV for pre-training. ER uses same SwAV pre-trained DNN as SIESTA. Additional details for the comparison algorithms are provided in Appendix B.

Table 2: Experimental results with state-of-the-art methods on ImageNet-1K. DER without pruning and DER with pruning are abbreviated as DER† and DER∗, respectively. The (↑) and (↓) indicate high and low values to reflect optimum performance respectively. $\mathcal{P}$ is the number of parameters in millions, μ is the average top-5 accuracy, α is the final top-5 accuracy, $\mathcal{M}$ is the total memory in GB, and $\mathcal{U}$ is the total number of updates in millions. We include SIESTA with (Aug) and without (No Aug) augmentations. DER∗ does not report $\mathcal{P}$, so we omit it.

Method	$\mathcal{P}(\downarrow)$	$\mu(\uparrow)$	$\alpha(\uparrow)$	$\mathcal{M}(\downarrow)$	$\mathcal{U}(\downarrow)$
End-to-End (Castro et al., 2018)	11.68	72.09	52.29	22.32	93.26
Simple-DER (Li et al., 2021)	28.00	85.62	80.76	22.39	213.17
iCaRL (Rebuffi et al., 2017)	11.68	63.70	44.00	22.32	79.94
BiC (Wu et al., 2019)	11.68	84.00	73.20	22.32	119.91
WA (Zhao et al., 2020)	11.68	86.60	81.10	22.32	133.23
DER† (Yan et al., 2021)	116.89	88.17	82.86	22.74	213.17
DER∗ (Yan et al., 2021)	—	87.08	81.89	22.32	213.17
DyTox (Douillard et al., 2022)	11.36	88.78	83.91	22.32	692.80
ER (Chaudhry et al., 2019)	5.48	82.19	71.22	19.59	58.78
REMIND (Gallardo et al., 2021)	11.68	83.61	77.14	2.05	58.78
SIESTA (No Aug)	5.48	88.33	83.59	2.02	**11.53**
SIESTA (Aug)	**5.48**	**90.67**	**87.00**	**2.02**	57.60

Overall results are in Table 2 and Figure 1. SIESTA performs best in average accuracy μ and final accuracy α, while having fewer DNN parameters $\mathcal{P}$, lower auxiliary memory usage $\mathcal{M}$, and fewer total updates $\mathcal{U}$. While REMIND uses the same amount of auxiliary memory and a similar number of DNN updates, SIESTA (Aug) outperforms REMIND by 9.86% (absolute) in final accuracy. SIESTA (Aug) exceeds DyTox, the method with the second highest final accuracy, by 3.09% (absolute), while using 12× fewer network updates and 11× less memory. SIESTA (No Aug) has comparable performance to state of the art batch learners like DER and DyTox, while requiring 18× fewer updates. Moreover, SIESTA (Aug) can provide even further performance gains (3.41% absolute improvement in final accuracy) at the cost of 5× more updates.

6.5 Ablation Studies

We conduct ablation experiments over a number of facets of SIESTA in order to gain insight into their relative importance. Details are in the Appendix.

Rehearsal Policies. In Appendix D, we compared eight different rehearsal policies. To do this, we used both ImageNet-1K and a challenging long-tailed dataset, Places-LT. On ImageNet, most policies performed similarly; however, large differences were seen among methods for Places-LT, where class balanced uniform sampling outperforms alternatives. SIESTA with or without balanced uniform rehearsal outperforms REMIND by 3.15% (uniform) and 7.88% (balanced uniform) in final accuracy on Places-LT. SIESTA achieves 4.73% higher final accuracy on Places-LT when using balanced uniform rehearsal compared to uniform rehearsal.

Architectures. We compare MobileNetV3-L and ResNet18 in Appendix E where MobileNetV3-L outperforms ResNet18 by 3.84% (augmentation) and 5.03% (no augmentation) in accuracy on ImageNet-1K.

Buffer Size. We study the impact of buffer size on SIESTA's performance in Appendix I where reducing buffer size from 2GB to 0.75GB results in only 3.02% absolute drop in final accuracy. Additionally, when both SIESTA and DER store same number of samples in buffer (130000 samples), SIESTA achieves 70.14% final accuracy on ImageNet-1K compared to DER's 70.15%. However, DER requires 10× more parameters and 95.4× more memory than SIESTA. When another variant of SIESTA, SIESTA-ER stores 130000 raw images like DER and performs veridical rehearsal, it outperforms DER with 71.86% final accuracy on ImageNet-1K. Furthermore, unlike DER, SIESTA is able to rival the offline model with a low memory footprint (see Table 1 and Table 6).

Does SIESTA Require Latent Rehearsal? For memory-efficiency, SIESTA uses latent rehearsal. To examine if SIESTA's innovations apply to the veridical rehearsal setting, where raw images are used, we created a veridical rehearsal variant of SIESTA. Compared to ER (Chaudhry et al., 2019), a widely used veridical rehearsal method,

this variant of SIESTA (SIESTA-ER) outperforms ER by 7.94% (absolute) in final accuracy on ImageNet-1K (see Figure 7), while being $3\times$ faster to train than ER on same hardware.

7 Discussion

This paper considers the problem of supervised continual learning, where the learner incrementally learns from a sequence about which it cannot make distributional assumptions. We argue that for real-world applications, continual learning needs to rival an offline learner and be more computationally efficient than periodically re-training from scratch. We also argue that systems need to be designed to handle arbitrary class orderings, where two extremes are iid and class-incremental learning, whereas many continual learning systems are bespoke to the incremental class learning scenario. We demonstrated that SIESTA largely meets these goals, achieving identical performance to the offline learner when augmentations are not used, and outperforming existing continual learning methods in accuracy using less compute when augmentations are used.

Computational efficiency in continual learning has long been a selling point of the research, but it has received little attention. Training large DNNs from scratch requires a huge amount of energy that can result in a large amount of greenhouse gas emissions (Luccioni et al., 2022; Patterson et al., 2021; Wu et al., 2022). From a financial perspective, training large models often requires a large amount of electricity, expensive hardware, and cloud computing resources. Many are calling for more computationally efficient algorithms to be developed (Strubell et al., 2020; Van Wynsberghe, 2021; Harun & Kanan, 2023), and we believe that continual learning can help address this problem while also enabling greater functionality provided by continuously updating DNNs with new information.

In this work, we only evaluated CNN architectures with SIESTA; however, the general framework is amenable to other architectures as long as the representations can be quantized. For example, SIESTA could be extended for models like Swin Transformers (Liu et al., 2021) or even Graph Neural Networks (Zhou et al., 2020), where we could quantize graph representations. Exploring the use of non-CNN architectures is critical for using SIESTA in non-vision modalities, e.g., audio and text data, which would be an exciting area of future work. Another interesting area of research could be incorporating additional data modalities over time to improve existing task performance and enhancing the learning of new tasks. Another direction is to use SIESTA for tasks such as continual learning in object detection, which was previously done using a REMIND-based system (Acharya et al., 2020).

SIESTA depends on the features learned in initial layers of the network, $\mathcal{H}$, being universal features for the domain, since they are not trained after the base initialization phase. Following others (Hayes et al., 2020; Belouadah & Popescu, 2019; Gallardo et al., 2021), we did base initialization on the first 100 classes of ImageNet-1K. While our model rivaled the offline learner when augmentations were not used, there was a small gap when augmentations were used between SIESTA and the offline learner. We hypothesize that this gap would be closed by improving the features in $\mathcal{H}$. Two potential ways to achieve this goal would be using a superior self-supervised learning algorithm than SwAV or by training on additional data. For continual learning for real applications, it would be prudent to initialize the DNN from a very large unlabeled dataset with self-supervised learning, which would likely work significantly better.

8 Conclusion

We proposed SIESTA, a scalable, faster and lightweight continual learning framework equipped with offline memory consolidation. We reduced computational overhead by making online learning free of rehearsal or expensive parametric updates during the wake phase. This closely aligns with real-time applications such as edge-devices, mobile phones, smart home appliances, robots, and virtual assistants. To effectively learn from a non-stationary data stream, the short-term wake memories were transferred into the DNN for long-term storage during offline sleep periods. Consequently, our model overcame forgetting of past knowledge. We showed that sleep improved online performance while outperforming state-of-the-art methods. SIESTA achieves similar accuracy to DyTox (Douillard et al., 2022) using an order of magnitude fewer updates when augmentations are not used, and surpasses DyTox in terms of accuracy when using augmentations. Although we evaluated SIESTA on image classification where augmentations are widely used, SIESTA can also be used for non-vision tasks where augmentations are not widely used.

References

Wickliffe C Abraham and Anthony Robins. Memory retention–the synaptic stability versus plasticity dilemma. *Trends in neurosciences*, 28(2):73–78, 2005.

Manoj Acharya, Tyler L Hayes, and Christopher Kanan. Rodeo: Replay for online object detection. In *BMVC*, 2020.

Rahaf Aljundi, Francesca Babiloni, Mohamed Elhoseiny, Marcus Rohrbach, and Tinne Tuytelaars. Memory aware synapses: Learning what (not) to forget. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 139–154, 2018.

Elahe Arani, Fahad Sarfraz, and Bahram Zonooz. Learning fast, learning slow: A general continual learning method based on complementary learning system. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=uxxFrDwrE7Y.

Eden Belouadah and Adrian Popescu. Il2m: Class incremental learning with dual memory. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 583–592, 2019.

Denise J Cai, Sarnoff A Mednick, Elizabeth M Harrison, Jennifer C Kanady, and Sara C Mednick. Rem, not incubation, improves creativity by priming associative networks. *Proceedings of the National Academy of Sciences*, 106 (25):10130–10134, 2009.

Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. *Advances in neural information processing systems*, 33: 9912–9924, 2020.

Francisco M Castro, Manuel J Marín-Jiménez, Nicolás Guil, Cordelia Schmid, and Karteek Alahari. End-to-end incremental learning. In *Proceedings of the European conference on computer vision (ECCV)*, pp. 233–248, 2018.

Arslan Chaudhry, Puneet K Dokania, Thalaiyasingam Ajanthan, and Philip HS Torr. Riemannian walk for incremental learning: Understanding forgetting and intransigence. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 532–547, 2018a.

Arslan Chaudhry, Marc'Aurelio Ranzato, Marcus Rohrbach, and Mohamed Elhoseiny. Efficient lifelong learning with a-gem. In *International Conference on Learning Representations*, 2018b.

Arslan Chaudhry, Marcus Rohrbach, Mohamed Elhoseiny, Thalaiyasingam Ajanthan, P Dokania, P Torr, and M Ranzato. Continual learning with tiny episodic memories. In *Workshop on Multi-Task and Lifelong Reinforcement Learning*, 2019.

William Jay Conover. *Practical nonparametric statistics*, volume 350. john wiley & sons, 1999.

Robert Coop, Aaron Mishtal, and Itamar Arel. Ensemble learning in fixed expansion layer networks for mitigating catastrophic forgetting. *IEEE transactions on neural networks and learning systems*, 24(10):1623–1634, 2013.

Rumen Dangovski, Li Jing, Charlotte Loh, Seungwook Han, Akash Srivastava, Brian Cheung, Pulkit Agrawal, and Marin Soljacic. Equivariant self-supervised learning: Encouraging equivariance in representations. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=gKLAAfiytI.

Prithviraj Dhar, Rajat Vikram Singh, Kuan-Chuan Peng, Ziyan Wu, and Rama Chellappa. Learning without memorizing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5138–5146, 2019.

Ina Djonlagic, Andrew Rosenfeld, Daphna Shohamy, Catherine Myers, Mark Gluck, and Robert Stickgold. Sleep enhances category learning. *Learning & memory*, 16(12):751–755, 2009.

Arthur Douillard, Alexandre Ramé, Guillaume Couairon, and Matthieu Cord. Dytox: Transformers for continual learning with dynamic token expansion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9285–9295, 2022.

Timothy J Draelos, Nadine E Miner, Christopher C Lamb, Jonathan A Cox, Craig M Vineyard, Kristofor D Carlson, William M Severa, Conrad D James, and James B Aimone. Neurogenesis deep learning: Extending deep networks to accommodate new classes. In *2017 International Joint Conference on Neural Networks (IJCNN)*, pp. 526–533. IEEE, 2017.

Simon J Durrant, Scott A Cairney, Cathal McDermott, and Penelope A Lewis. Schema-conformant memories are preferentially consolidated during rem sleep. *Neurobiology of Learning and Memory*, 122:41–50, 2015.

Zhiyuan Fang, Jianfeng Wang, Lijuan Wang, Lei Zhang, Yezhou Yang, and Zicheng Liu. Seed: Self-supervised distillation for visual representation. In *International Conference on Learning Representations*, 2020.

Chrisantha Fernando, Dylan Banarse, Charles Blundell, Yori Zwols, David Ha, Andrei A Rusu, Alexander Pritzel, and Daan Wierstra. Pathnet: Evolution channels gradient descent in super neural networks. *arXiv preprint arXiv:1701.08734*, 2017.

Robert M French. Pseudo-recurrent connectionist networks: An approach to the'sensitivity-stability'dilemma. *Connection Science*, 9(4):353–380, 1997.

Jhair Gallardo, Tyler L Hayes, and Christopher Kanan. Self-supervised training enhances online continual learning. *In British Machine Vision Conference (BMVC)*, 2021.

Siavash Golkar, Michael Kagan, and Kyunghyun Cho. Continual learning via neural pruning. *arXiv preprint arXiv:1903.04476*, 2019.

Hasan Abed Al Kader Hammoud, Ameya Prabhu, Ser-Nam Lim, Philip HS Torr, Adel Bibi, and Bernard Ghanem. Rapid adaptation in online continual learning: Are we evaluating it right? *arXiv preprint arXiv:2305.09275*, 2023.

Md Yousuf Harun and Christopher Kanan. Overcoming the stability gap in continual learning. *arXiv preprint arXiv:2306.01904*, 2023.

Md Yousuf Harun, Jhair Gallardo, Tyler L Hayes, and Christopher Kanan. How efficient are today's continual learning algorithms? In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2430–2435, 2023.

Tyler L Hayes and Christopher Kanan. Lifelong machine learning with deep streaming linear discriminant analysis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pp. 220–221, 2020.

Tyler L Hayes and Christopher Kanan. Online continual learning for embedded devices. In *CoLLAs*, 2022.

Tyler L Hayes, Nathan D Cahill, and Christopher Kanan. Memory efficient experience replay for streaming learning. In *2019 International Conference on Robotics and Automation (ICRA)*, pp. 9769–9776. IEEE, 2019.

Tyler L Hayes, Kushal Kafle, Robik Shrestha, Manoj Acharya, and Christopher Kanan. Remind your neural network to prevent catastrophic forgetting. In *European Conference on Computer Vision*, pp. 466–483. Springer, 2020.

Tyler L Hayes, Giri P Krishnan, Maxim Bazhenov, Hava T Siegelmann, Terrence J Sejnowski, and Christopher Kanan. Replay in deep learning: Current approaches and missing biological elements. *Neural Computation*, 33(11):2908–2950, 2021.

P Hetherington. Is there'catastrophic interference'in connectionist networks? In *Proceedings of the 11th annual conference of the cognitive science society*, pp. 26–33. LEA, 1989.

Saihui Hou, Xinyu Pan, Chen Change Loy, Zilei Wang, and Dahua Lin. Lifelong learning via progressive distillation and retrospection. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 437–452, 2018.

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

Andrew Howard, Mark Sandler, Grace Chu, Liang-Chieh Chen, Bo Chen, Mingxing Tan, Weijun Wang, Yukun Zhu, Ruoming Pang, Vijay Vasudevan, et al. Searching for mobilenetv3. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 1314–1324, 2019.

Ching-Yi Hung, Cheng-Hao Tu, Cheng-En Wu, Chien-Hung Chen, Yi-Ming Chan, and Chu-Song Chen. Compacting, picking and growing for unforgetting continual learning. *Advances in Neural Information Processing Systems*, 32, 2019.

Herve Jegou, Matthijs Douze, and Cordelia Schmid. Product quantization for nearest neighbor search. *IEEE transactions on pattern analysis and machine intelligence*, 33(1):117–128, 2010.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. Billion-scale similarity search with gpus. *IEEE Transactions on Big Data*, 2019.

Minsoo Kang, Jaeyoo Park, and Bohyung Han. Class-incremental learning by knowledge distillation with adaptive feature consolidation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 16071–16080, 2022.

Ronald Kemker and Christopher Kanan. Fearnet: Brain-inspired model for incremental learning. In *ICLR*, 2018.

Ronald Kemker, Marc McClure, Angelina Abitino, Tyler Hayes, and Christopher Kanan. Measuring catastrophic forgetting in neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2018.

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

Simon Kornblith, Ting Chen, Honglak Lee, and Mohammad Norouzi. Why do better loss functions lead to less transferable features? *Advances in Neural Information Processing Systems*, 34:28648–28662, 2021.

Dong Hoon Lee, Sungik Choi, Hyunwoo J Kim, and Sae-Young Chung. Unsupervised visual representation learning via mutual information regularized assignment. *Advances in Neural Information Processing Systems*, 35:29610–29623, 2022.

Penelope A. Lewis, Günther Knoblich, and Gina Poe. How Memory Replay in Sleep Boosts Creative Problem-Solving. *Trends in Cognitive Sciences*, 22(6):491–503, June 2018. ISSN 13646613. doi: 10.1016/j.tics.2018.03.009. URL http://linkinghub.elsevier.com/retrieve/pii/S1364661318300706.

Wei Li, Lei Ma, Guang Yang, and Wen-Biao Gan. Rem sleep selectively prunes and maintains new synapses in development and learning. *Nature neuroscience*, 20(3):427–437, 2017.

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

Zhuoyun Li, Changhong Zhong, Sijia Liu, Ruixuan Wang, and Wei-Shi Zheng. Preserving earlier knowledge in continual learning with the help of all previous feature extractors. *arXiv preprint arXiv:2104.13614*, 2021.

Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 10012–10022, 2021.

Ziwei Liu, Zhongqi Miao, Xiaohang Zhan, Jiayun Wang, Boqing Gong, and Stella X Yu. Large-scale long-tailed recognition in an open world. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 2537–2546, 2019.

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

Alexandra Sasha Luccioni, Sylvain Viguier, and Anne-Laure Ligozat. Estimating the carbon footprint of bloom, a 176b parameter language model. *arXiv preprint arXiv:2211.02001*, 2022.

James L McClelland and Nigel H Goddard. Considerations arising from a complementary learning systems perspective on hippocampus and neocortex. *Hippocampus*, 6(6):654–665, 1996.

Thomas Mensink, Jakob Verbeek, Florent Perronnin, and Gabriela Csurka. Distance-based image classification: Generalizing to new classes at near-zero cost. *IEEE transactions on pattern analysis and machine intelligence*, 35(11): 2624–2637, 2013.

Oleksiy Ostapenko, Mihai Puscas, Tassilo Klein, Patrick Jahnichen, and Moin Nabi. Learning to remember: A synaptic plasticity driven framework for continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11321–11329, 2019.

Pingbo Pan, Siddharth Swaroop, Alexander Immer, Runa Eschenhagen, Richard Turner, and Mohammad Emtiyaz E Khan. Continual deep learning by functional regularisation of memorable past. *Advances in Neural Information Processing Systems*, 33:4453–4464, 2020.

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

David Patterson, Joseph Gonzalez, Quoc Le, Chen Liang, Lluis-Miquel Munguia, Daniel Rothchild, David So, Maud Texier, and Jeff Dean. Carbon emissions and large neural network training. *arXiv preprint arXiv:2104.10350*, 2021.

Quang Pham, Chenghao Liu, and Steven Hoi. Dualnet: Continual learning, fast and slow. *Advances in Neural Information Processing Systems*, 34:16131–16144, 2021.

Quang Pham, Chenghao Liu, Doyen Sahoo, and Steven Hoi. Learning fast and slow for time series forecasting. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=q-PbpHD3EOk

Ameya Prabhu, Hasan Abed Al Kader Hammoud, Puneet K Dokania, Philip HS Torr, Ser-Nam Lim, Bernard Ghanem, and Adel Bibi. Computationally budgeted continual learning: What does matter? In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3698–3707, 2023a.

Ameya Prabhu, Zhipeng Cai, Puneet Dokania, Philip Torr, Vladlen Koltun, and Ozan Sener. Online continual learning without the storage constraint. *arXiv preprint arXiv:2305.09253*, 2023b.

Siyuan Qiao, Huiyu Wang, Chenxi Liu, Wei Shen, and Alan Yuille. Micro-batch training with batch-channel normalization and weight standardization. *arXiv preprint arXiv:1903.10520*, 2019.

Aljundi Rahaf and Caccia Lucas. Online continual learning with maximally interfered retrieval. In *NIPS*, 2019.

Jathushan Rajasegaran, Munawar Hayat, Salman H Khan, Fahad Shahbaz Khan, and Ling Shao. Random path selection for continual learning. *Advances in Neural Information Processing Systems*, 32, 2019.

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

Hippolyt Ritter, Aleksandar Botev, and David Barber. Online structured laplace approximations for overcoming catastrophic forgetting. *Advances in Neural Information Processing Systems*, 31, 2018.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015.

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

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

Carlyle Smith and Danielle Smith. Ingestion of ethanol just prior to sleep onset impairs memory for procedural but not declarative tasks. *Sleep*, 26(2):185–191, 2003.

Leslie N Smith and Nicholay Topin. Super-convergence: Very fast training of neural networks using large learning rates. In *Artificial intelligence and machine learning for multi-domain operations applications*, volume 11006, pp. 369–386. SPIE, 2019.

Ben Sorscher, Robert Geirhos, Shashank Shekhar, Surya Ganguli, and Ari Morcos. Beyond neural scaling laws: beating power law scaling via data pruning. *Advances in Neural Information Processing Systems*, 35:19523–19536, 2022.

Emma Strubell, Ananya Ganesh, and Andrew McCallum. Energy and policy considerations for modern deep learning research. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 13693–13696, 2020.

Fuwen Tan, Fatemeh Sadat Saleh, and Brais Martinez. Effective self-supervised pre-training on low-compute networks without distillation. In *The Eleventh International Conference on Learning Representations*, 2022.

Xiaoyu Tao, Xinyuan Chang, Xiaopeng Hong, Xing Wei, and Yihong Gong. Topology-preserving class-incremental learning. In *European Conference on Computer Vision*, pp. 254–270. Springer, 2020.

Michalis K Titsias, Jonathan Schwarz, Alexander G de G Matthews, Razvan Pascanu, and Yee Whye Teh. Functional regularisation for continual learning with gaussian processes. In *International Conference on Learning Representations*, 2019.

Aimee Van Wynsberghe. Sustainable ai: Ai for sustainability and the sustainability of ai. *AI and Ethics*, 1(3):213–218, 2021.

Vikas Verma, Alex Lamb, Christopher Beckham, Amir Najafi, Ioannis Mitliagkas, David Lopez-Paz, and Yoshua Bengio. Manifold mixup: Better representations by interpolating hidden states. In *International conference on machine learning*, pp. 6438–6447. PMLR, 2019.

Albrecht P Vorster and Jan Born. Sleep and memory in mammals, birds and invertebrates. *Neuroscience & Biobehavioral Reviews*, 50:103–119, 2015.

Kai Wang, Joost van de Weijer, and Luis Herranz. Acae-remind for online continual learning with compressed feature replay. *Pattern Recognition Letters*, 150:122–129, 2021.

Carole-Jean Wu, Ramya Raghavendra, Udit Gupta, Bilge Acun, Newsha Ardalani, Kiwan Maeng, Gloria Chang, Fiona Aga, Jinshi Huang, Charles Bai, et al. Sustainable ai: Environmental implications, challenges and opportunities. *Proceedings of Machine Learning and Systems*, 4:795–813, 2022.

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

Shipeng Yan, Jiangwei Xie, and Xuming He. Der: Dynamically expandable representation for class incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3014–3023, 2021.

J Yoon, E Yang, J Lee, and SJ Hwang. Lifelong learning with dynamically expandable networks. International Conference on Learning Representations, ICLR, 2018.

Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 6023–6032, 2019.

Jure Zbontar, Li Jing, Ishan Misra, Yann LeCun, and Stéphane Deny. Barlow twins: Self-supervised learning via redundancy reduction. In *International Conference on Machine Learning*, pp. 12310–12320. PMLR, 2021.

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

Bowen Zhao, Xi Xiao, Guojun Gan, Bin Zhang, and Shu-Tao Xia. Maintaining discrimination and fairness in class incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 13208–13217, 2020.

Bolei Zhou, Agata Lapedriza, Aditya Khosla, Aude Oliva, and Antonio Torralba. Places: A 10 million image database for scene recognition. *IEEE transactions on pattern analysis and machine intelligence*, 40(6):1452–1464, 2017.

Jie Zhou, Ganqu Cui, Shengding Hu, Zhengyan Zhang, Cheng Yang, Zhiyuan Liu, Lifeng Wang, Changcheng Li, and Maosong Sun. Graph neural networks: A review of methods and applications. *AI open*, 1:57–81, 2020.