Batch Model Consolidation: A Multi-Task Model Consolidation Framework

Iordanis Fostiropoulos Jiaye Zhu Laurent Itti
University of Southern California, Los Angeles, United States
{fostirop, jiayezhu, itti}@usc.edu

Abstract

*In Continual Learning (CL), a model is required to learn a stream of tasks sequentially without significant performance degradation on previously learned tasks. Current approaches fail for a long sequence of tasks from diverse domains and difficulties. Many of the existing CL approaches are difficult to apply in practice due to excessive memory cost or training time, or are tightly coupled to a single device. With the intuition derived from the widely applied mini-batch training, we propose Batch Model Consolidation (**BMC**) to support more realistic CL under conditions where multiple agents are exposed to a range of tasks. During a regularization phase, BMC trains multiple expert models in parallel on a set of disjoint tasks. Each expert maintains weight similarity to a base model through a stability loss, and constructs a buffer from a fraction of the task's data. During the consolidation phase, we combine the learned knowledge on 'batches' of expert models using a batched consolidation loss in memory data that aggregates all buffers. We thoroughly evaluate each component of our method in an ablation study and demonstrate the effectiveness on standardized benchmark datasets Split-CIFAR-100, Tiny-ImageNet, and the Stream dataset composed of 71 image classification tasks from diverse domains and difficulties. Our method outperforms the next best CL approach by 70% and is the only approach that can maintain performance at the end of 71 tasks.*

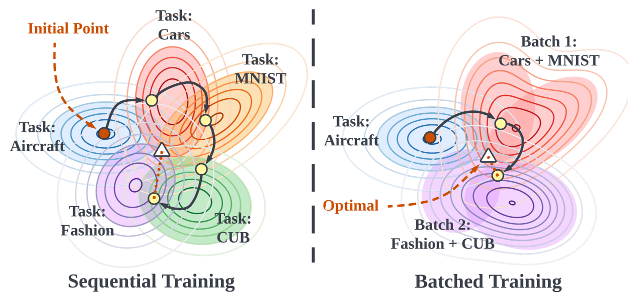

Figure 1. The loss contours by sequential training compared with batch task training [1] (shaded areas as low-error zones for each task). Intuition: Similar to mini-batch training batched task training can reduce the local minima and improve the convexity of the loss landscape.

1. Introduction

Continual Learning (CL) has allowed deep learning models to learn in a real world that is constantly evolving, in which data distributions change, goals are updated, and critically, much of the information that any model will encounter is not immediately available [2]. Current approaches in CL provide a trade-off to the stability-plasticity dilemma [3] where improving performance for a novel task leads to catastrophic forgetting.

Continual Learning benchmarks are composed of a limited number of tasks and with tasks of non-distinct domains, such as Split-CIFAR100 [4] and Split-Tiny-ImageNet [5]. Previous approaches in Continual Learning suffer significant performance degradation when faced with a large number of tasks, or tasks from diverse domains [1]. Additionally, the cost of many methods increases with the number of tasks [6,7] and becomes ultimately unacceptable for certain applications, while other methods [8–11] are tightly coupled to training on a single device and therefore cannot benefit from scaling in distributed settings. As such, current approaches are impractical for many real-world applications, where multiple devices are trained on a diverse and disjoint set of tasks with the goal of maintaining a single model.

Motivated by the performance, memory cost, training time and flexibility issues of current approaches, we propose **Batch Model Consolidation (BMC)**, a Continual Learning framework that supports distributed training on multiple streams of diverse tasks, but also improves performance when applied on a single long task stream. Our method trains and consolidates multiple workers that each become an expert in a task that is disjoint from all other tasks. In contrast, for Federated Learning the training set is composed of a single task of heterogeneous data [12].

Our method is composed of two phases. First, during the **regularization phase**, a set of *expert models* is trained in new tasks in parallel with their weights regularized to a

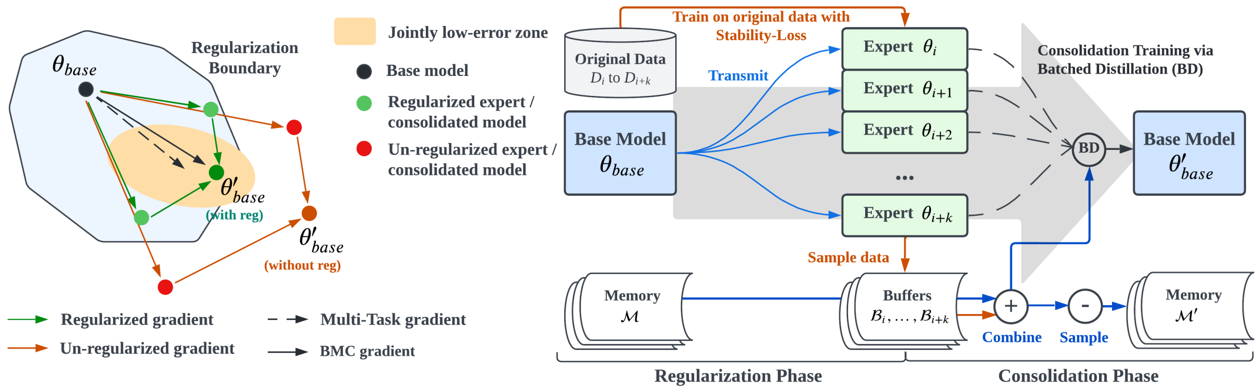

Figure 2. A single incremental step of BMC. On the **right** figure, the updating of a base model with *Multi-Expert Training*: after receiving the data of the new tasks $D_i, \dots, D_{i+k}$, a batch of experts $\theta_i, \dots, \theta_{i+k}$ are trained separately on their corresponding tasks with *stability loss* applied from the base model. The newly trained experts then sample a subset of their training data and combine them with the memory to perform *batched distillation* on the base model. On the **left** figure, the regularization helps the batched distillation to update the model closer to the regularization boundary and towards the jointly low-error zone of old tasks and two new tasks.

base model. Second, during the **consolidation phase** the expert models are combined into the base model in a way that better retains the performance on the current tasks of all experts and all previously learned tasks. The main advantage of our method is that it provides a better approximation to the *multi-task gradient* of all tasks from all expert models, Fig. 2. Lastly, BMC better retains performance for significantly more tasks than current baselines, while reducing the total time of training when compared to training on the same task-stream in a sequential manner. The primary contributions of our paper are as follows.

1. We propose Batch Model Consolidation (**BMC**) to support CL for training multiple *expert models* on a single task stream composed of tasks from diverse domains.
2. We extend *BMC* for a distributed learning framework where we train multiple *expert models* on disjoint task streams.
3. We propose a *stability loss* to reduce forgetting that is applied between *expert models* and a *base model*. Lastly, a *batched distillation loss* combines multiple *expert models* to update a single *base model* in a single incremental step.
4. We verify our approach on popular benchmarks and show that BMC is robust against large domain-shifts and for a large number of tasks.

2. Related Works

Following the taxonomy by De Lange *et al*. [13] we summarize methods that mitigate forgetting in three categories, Replay, Regularization and Parameter isolation methods.

Replay methods identify a limited number of exemplars to store in an auxiliary dataset, *buffer*, that is used to retain performance on previously seen tasks through *rehearsal* (ER [14], GEM [15], A-GEM [16], GSS [17]). An auxiliary loss can be applied as a regularization term to the main training task, such as with *Knowledge Distillation* (DER++ [18], iCaRL [19], FDR [20], DMC [21], ExModel [22]) or by restricting the gradient magnitude (GEM [15], A-GEM [16]). Exemplars can be randomly selected from the original dataset [18] or synthetically generated [22]. Similarly, we perform distillation on previously stored exemplars in a *memory* bank to consolidate knowledge from previous tasks. In contrast to previous works we perform a two-step process of a *regularization-phase* where we maintain proximity of the newly trained task-specific (expert) model to the old (base) model by a *stability loss* as opposed to 'knowledge transfer', and in the second *consolidation-phase*, we apply *batched distillation loss* on the pair-wise intermediate representations between **multiple** expert models and an older model on real exemplars from a buffer. We identify that combining multiple 'teacher' models in a single step is better than a single 'teacher' model in multiple steps (DMC [21]), and we provide a theoretical justification of the result in Sec. 3.

Other methods store exemplars in their buffer using prototypes [19], increasing exemplar representability [17], or gradient projection [15, 16] to restrict drastic gradient updates. Such methods are orthogonal to our method, since BMC is extendable to different types of buffer sampling methods and regularization loss as the *stability loss*.

Parameter-isolation approaches keep the important weights fixed to reduce forgetting. SupSup [23], HAT [24], PSP [25], PNN [7], and BLIP [26] identify and assign task-specific parameters in the model via *supermasks* or by ap-

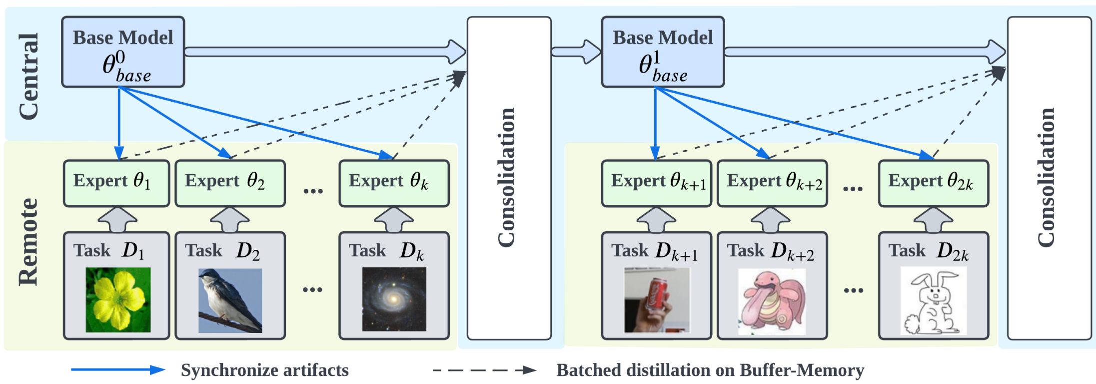

Figure 3. Distributed training paradigm of BMC. The central device maintains only a base model and Memory. All experts can be trained in parallel on non-overlapping tasks using distributed devices. Batched Distillation is applied agnostic to the expert model weights on Buffer and Memory data.

pending new weights to the model [7]. Model Zoo [27] infers and trains a group of similar tasks into one model as a 'weak classifier' to utilize shared domain knowledge, and use an ensemble of models during inference. Such approaches have the number of parameters grow with respect to the number of tasks. Methods such as PackNet [28] and RMN [29], overwrite unimportant parameters to provide larger model capacity for new tasks and do not grow indefinitely. Similarly, we assign an *expert model* to each task to isolate the parameters. However, we maintain the number of experts at each incremental step fixed so that the cost of our method remains constant as the number of tasks grows. Finally, we perform inference using a single *base* model as opposed to an ensemble of models.

Regularization methods such as EWC [8] and similarly (MAS [30], SI [11]) use an auxiliary loss term to constrain optimization w.r.t. to a metric of importance for each parameter for a given task. LwF [31] distills knowledge from the previous model using current task data, and LFL [9] freezes portion of the network while penalizing intermediate representations using the Euclidean distance. These approaches are orthogonal to our method and are candidates for the *stability loss*. We find that they underperform compared to our method in Sec. 5.2 and additional experiments in the supplementary.

Distributed Continual Learning combines Continual Learning and Federated Learning by incrementally learning a model stored on a central server using distributed devices. Previous works learn the same task on multiple distributed devices that are allocated different subsets of data (CFeD [32]); or train the same sequence of tasks on each device with inter-client communication of model parameters (FedWeIT [33]). Our method also combines the knowledge of multiple models trained on distributed devices. In contrast to previous work, we train a unique task on each device, to learn an *expert model*. We consolidate multiple expert models with *batched distillation* on real data as opposed to Knowledge Distillation on an auxiliary dataset [32] or simple aggregation [33]. Lastly, in contrast to Federated Learning setting, our method addresses performance constraints as opposed to privacy constraints and prohibits inter-client communication. The remote devices communicate once per incremental step with the central server.

In summary, our approach is a combination of a Regularization method with the use of the stability loss, Replay method with the use of batched distillation loss, and finally Parameter-isolation where we train multiple experts on disjoint tasks and in a distributed fashion.

3. Preliminaries

Continual Learning aims to learn a new task from a stream of sequential tasks and without access to previous task datasets, while maintaining performance on all previous tasks. Given a set of Tasks $T = \{T_1 \dots T_n\}$, with clear task boundaries. We train and evaluate our method in a **Class Incremental Learning (CIL)** [2, 6] setting in which task identity is provided during training but not at test time. The Continual Learning objective function for a model with parameters θ, can be summarized as maximizing the average classification accuracy after learning a sequence of T tasks:

$$\overline{A} = \max_{\theta} \frac{1}{T} \sum_{t=1}^{T} Acc(\theta(T_i), T_{i_y}) \tag{1}$$

Knowledge Distillation (KD) [34], in the CL setting, can be used to transfer knowledge between models trained on different tasks [18, 19]. KD penalizes the student model using a loss function between the representations of teacher and student models. Representations used in KD can be the output logits $\mathcal{L}_{kd}$ [18, 19, 21, 22] or the intermediate feature vectors $\mathcal{L}_{bd}$ [35, 36]. Given the hidden representation vectors at depth $i \in |\theta|$ from student and teacher models, $\phi_i^s, \phi_i^t \in \mathbb{R}^d$, we compute $\mathcal{L}_{bd}$ as the sum of the distance between all pair-wise hidden representations.

$$\mathcal{L}_{bd}(\theta_t(x), \theta_s(x)) = \sum_{i=1}^{|\theta|} ||sg(\phi_i^t) - \phi_i^s||_2 \tag{2}$$

sg is the stop gradient operator that prevents the parameters of the teacher model from being updated.

Gradient Noise Reduction In many nonconvex optimization problems, the loss manifold is filled with local minima and saddle points, where Stochastic Gradient Descent (SGD) optimization can underperform [37]. Noise in the training data leads to noise in the gradients and high variance for SGD [38]. Gradient approximation methods, such as minibatch training, accumulate gradients over a batch of data to estimate the *true gradient* of the entire training set. Keskar *et al*. [39] observed that as the batch becomes smaller, the parameters are updated further away from their initial point as opposed to large batch training. This observation is in agreement with [38, 40] that small batch introduces more 'randomness' as it is a lesser approximation to the true gradient of the entire training set and causes instability in training.

Similarly, consider a continuum learning environment where there is a set of *expert* models trained on a disjoint set of tasks with the goal of consolidating them sequentially into a single *base* model. We argue that the consolidation training process is similar to batch training but in the multi-task setting, where we reduce the variance by consolidating the experiences from multiple experts. Previous model consolidation methods [21, 22] combine a single expert at a time. In contrast, for our method, we observe that batched consolidation improves accuracy as well as enables data parallelism that can speed up training.

4. Method

Batch Model Consolidation (BMC), combines a rehearsal-based learning system and a two-step training process, Fig. 2 (right). We first introduce the main design components of BMC for a single task sequence, a *task stream*. Next, we formalize the constraints under which we evaluate *Multi-Expert Training*, where multiple experts are trained in parallel on distinct *task streams* composed of sequences with distinct tasks. In short, our method is composed of multiple *training incremental steps* until all tasks are learned. Each step trains multiple expert models in parallel. Each expert is trained on a specific task, different from all other tasks. The training of each expert is composed of a **regularization phase** that reduces the deviation of the parameters for the current task from the base model. At the end of the training of all expert models for the current incremental step, a **consolidation phase** distills the expert knowledge back to the base model simultaneously using **batched distillation loss**. Our method performs better as compared to single-model distillation, and is a better approximation to the 'multi-task gradient' Fig. 2 (left) and Fig. 1.

4.1. Buffer-Memory

BMC uses a short-term buffer storage and a long-term memory bank to store real samples for rehearsal. **Memory** is a fixed-size storage that holds training exemplars from multiple previous tasks and is only accessible by the base model. **Buffer** is a temporary storage of limited size for a subset of the expert's training data. For each *train incremental step* of k experts, at the end of the *regularization phase*, the central memory bank is combined with $\mathcal{B}_1$, ..., $\mathcal{B}_k$ from experts. At the end of the *consolidation phase*, the memory is subsampled to maintain a constant size.

Sampling methods are applied for both memory and buffer data selection to meet the size constraints of each storage solution. The goal of a sampling method is to improve the informativeness of the buffer data for the current task and the memory data for all previously learned tasks. We experiment with multiple sampling methods, including gradient-based sampling [17] and random selection in an ablation study Sec. 5.2.

4.2. Stability Loss

During the Regularization phase, we train an **expert** model θ_{exp} that is initialized from a **base** model θ_{base}. Stability loss is applied during the regularization phase and poses a constraint optimization problem during the training of the expert on the new task. The goal of the expert is to learn the new task while maintaining feature similarity to previously learned tasks represented by θ_{base}, implicitly and without access to previous task data. The idea follows a direct comparison to previous Continual Learning regularization-based approaches [11, 30, 31, 41]. The intuition of the stability loss is to make the model less prone to *task-recency bias* [2] which can be viewed as the root cause of forgetting. Additionally, our ablation studies support the view that the stability loss improves consolidation as each expert's weights are confined within the regularization boundary (Fig. 2). The optimization objective of each expert can be summarized as:

$$\mathcal{L}_{exp} = \mathcal{L}_T(\theta_{exp}(x), y) + \lambda \mathcal{L}_{bd}(\theta_{base}(x), \theta_{exp}(x)) \tag{3}$$

where $\mathcal{L}_T$ is the task loss and $\mathcal{L}_{bd}$ is the distillation loss applied with the base model as the 'teacher' and the 'expert' model as the student and λ is the stability coefficient. EWC, or KD on the logits can be applied in direct replacement to $\mathcal{L}_{bd}$. We find experimentally that they under-perform compared to $\mathcal{L}_{bd}$ and discuss the ablation experiment results in Sec. 5 with additional experiments in the supplementary.

4.3. Consolidation Phase

At the end of training and for all k experts, BMC consolidates the learned knowledge of all experts in a single training step using a batched distillation loss. Batched distillation loss is applied only to the most recent task buffer data $\mathcal{B}$ and to memory $\mathcal{M}$ such that $\mathcal{D} = \{\mathcal{M}, \mathcal{B}_1, \ldots, \mathcal{B}_k\}$.

Instead of performing Knowledge Distillation on a single model or task at a time, batched distillation loss is applied with randomly sampled buffer data and expert representations from $\mathcal{D}$. As such, each training batch for the base model can contain randomly sampled tasks from multiple domains. We hypothesize that batched distillation loss, $\mathcal{L}_{bmc}$, improves performance by improving the convexity of the loss landscape for all tasks, Fig. 1. $\mathcal{L}_{bmc}$ penalizes on the difference in feature representations from θ'_{base} to all experts $\mathcal{E} = \{\theta_1, \ldots, \theta_k\}$.

$$\mathcal{L}_{bmc} = \sum_{\theta_i \in \mathcal{E}} \mathbb{E}_{x, \phi(x;\theta_i) \sim \mathcal{D}} [\mathcal{L}_{bd}(\theta'_{base}(x), \theta_i(x))] \tag{4}$$

We experiment with alternatives to $\mathcal{L}_{bd}$ when computing $\mathcal{L}_{bmc}$. We find that $\mathcal{L}_{bd}$ performs the best with experimental results in Sec. 5 and the supplementary. The final optimization objective of the base model is the joint optimization of $\mathcal{L}_{bmc}$ and the *experience replay* task loss. The training objective of the base model can be summarized as:

$$\mathcal{L}_{base} = \alpha \mathcal{L}_T(\theta'_{base}(x), y) + \beta \mathcal{L}_{bmc}(\theta'_{base}, \mathcal{D}) \tag{5}$$

where $(x, y) \in \mathcal{D}$, α is the experience replay task loss coefficient and β the consolidation coefficient.

4.4. Multi-Expert Training

We formalize the storage and communication constraints under which we evaluate our method. Multi-Expert Training involves multiple devices trained on disjoint tasks with the goal of learning and maintaining a single model that can perform well on all tasks. Thus, a method is evaluated on the cost under which it can mitigate forgetting and penalizes methods that can grow indefinitely with the number of tasks.

Each train incremental step begins with a *synchronization* phase where the base model initializes each expert with the weights of the base model θ_{base}. Next, and during the *consolidation* phase, the expert models communicate the consolidation artifacts to the central device. We evaluate and report the performance of the updated base model at the end of the *consolidation phase*.

What is shared between each Expert and the Base model is a flexible design choice that is penalized by the **Communication cost** $\mathbb{B}_c$, as the total number of bytes sent between the central and remote devices in each incremental training step and the **Memory cost** $\mathbb{M}_c$ of data maintained on the central device. We use **Cost Accuracy** $\mathbb{A}_c$ to evaluate a

---

Algorithm 1 Distributed Batch Model Consolidation

Central Device:

Initialize base model θ_{base} and memory $\mathcal{M}$
for batch of k tasks $\in T$ **do**
  Synchronization phase Transmit θ_{base}
  Run Remote Train
  Consolidation phase Transmit $\mathbb{B}$
  Train θ'_{base} using $\mathbb{B}$ by Eq. (5)
  $\mathcal{M}' \leftarrow$ sample$(\mathcal{M} \cup \mathcal{B})$
  Discard $\mathbb{B}$
end for

Remote Train:

Given current task dataset T_i and θ_{base}
Initialize expert $\theta_i \leftarrow \theta_{base}$
Train θ_i on T_i by Eq. (3)
Sample $\mathcal{B}_i$ from T_i
Transmit $\mathcal{B}_i$

---

method, on the marginal rate of substitution of mean accuracy, 1 to the **Total Cost** $\mathbb{T}_c$ at the given evaluation point

$$\mathbb{A}_c = \frac{\overline{A}}{\mathbb{T}_c} \tag{6}$$

where $\mathbb{T}_c = \overline{\mathbb{M}_c \mathbb{B}_c} + \theta$. In detail, for a given train incremental step, we pose the following constraints:

- Each expert θ_i represents a remote device that has access only to the current task data T_i and θ_{base}.
- Each remote device can communicate once at the end of the training process to the central device consolidation artifacts $\mathbb{B}$.
- The central device must use the consolidation artifacts and $\mathcal{M}$ to update θ'_{base}

Thus, the constraints use $\mathbb{M}_c$ to penalize a method that can naively isolate parameters for each task or store all train artifacts at the end of a consolidation phase. Furthermore, the constraints use $\mathbb{B}_c$ to penalize methods that can transmit a buffer that is similar in size to the task dataset on which the expert was trained. The communication constraints can provide a Pareto front of a method in how well it utilizes the available data. As a real-world example, consider the scenario where a fleet of autonomous vehicles are trained on geographical regions of diverse and disjoint features [42], such as weather patterns and road conditions. Additionally, communication of training data can be prohibitively expensive, and storage of Buffer data or models infeasible for a large fleet of vehicles. Fig. 3 provides an illustration of the Multi-Expert training and Algorithm 1 the pseudo-code for our method.

5. Experiments

We evaluate our method on three Continual Learning benchmarks, Tiny-ImageNet [5] split into 10 tasks, CIFAR-100 [4] split into 10 tasks and 20 tasks. Next, we evaluate our method on a long sequence of diverse tasks to demonstrate BMC's advantage. We use the Stream dataset composed of 71 image classification tasks for rigorous evaluation on average accuracy, Cost Accuracy, and relative training time. Lastly, we evaluate the efficacy of each design component for our method through ablation experiments on Permuted-MNIST, where we train for 128 tasks and a total of 1280 classes. We provide an overview of the dataset in this section and attach additional details in the supplementary. We open-source and provide the extracted feature vectors from the Stream dataset, the code to run the benchmark on the baselines and the code for our method as a Distributed Continual Learning library[1].

Stream Dataset. Common benchmarks for evaluating Continual Learning methods are built by splitting classes from datasets such as MNIST, CIFAR-10/100 and Tiny-ImageNet, which have subtasks in similar domains, of similar size and number of classes. We aim to evaluate BMC in a more general setting where there are larger domain-shifts, for significantly more tasks that range in difficulty and problem size. Lastly, synthetically generated datasets such as permuted-MNIST can be poor references to performance in applicable scenarios [6]. To this end, we use Stream which is composed of 71 publicly available image classification datasets [43–113] from the computer vision literature and Kaggle [114]. We concatenate the datasets into a stream of tasks. There are a total of 6,770,722 train images and 743,977 validation images with 2866 classes in Stream, with different numbers of classes for each task. Details on each dataset are attached in the supplementary. To speed up the experiments, we extract feature vectors from a pre-trained CLIP model [115] and used them as input to the model. For both our method and the baselines we use identical train hyper-parameters. We use the hyper-parameters as reported in the original paper for each method, where it is applicable. All experiments use an MLP with Residual connections [116] on the extracted CLIP feature vectors.

Baselines. We follow the methodology and compare with the methods reported in [6, 18]. When running a method on the Stream dataset we use implementation by Mammoth [18] and FACIL [2]. We report results from each respective paper when they are available or work by [6,18]. We compare our method with ER [14], GSS [17], A-GEM [16], iCaRL [19], GDumb [117], DER++ [18], Online EWC [41], SI [11], MAS [30] and DMC [21] with details of each method on Sec. 2. We train a naive baseline (*SGD*) without any Continual Learning strategy as a lower-bound. We compute the theoretical upper bound on performance as *Multi-Task* accuracy where we use the mean accuracy of SGD on each task.

[1] https://github.com/fostiropoulos/stream_benchmark

5.1. Results

We use 10 experts for our experiments (other training configurations are provided in Supplementary). Table 1 report the average accuracy $\overline{A}$, for the main baselines. We report Cost Accuracy $\mathbb{A}_c$, Total Cost $\mathbb{T}_c$ and relative training time to SGD. We include additional baselines in Fig. 4 and a full table in the supplementary. For the Continual Learning benchmarks, we show that BMC works well on the short sequence of similar tasks. For the Stream benchmark, our method significantly outperforms all baselines. In detail, BMC outperforms the second highest (ER) by 70%, and achieve 79% of the theoretical upper bound provided by Multi-Task training. Additionally, BMC has a constant time complexity w.r.t. the number of previously seen tasks and is 22% faster when compared to training with SGD on a single device. Other baselines degrade in relative time performance because they require a second backward propagation [18] or have an intractable training time as the number of learned tasks increases [17, 18, 118].

Our experiments and benchmark conclude that most of the recent approaches fail in mitigating forgetting and are outperformed in all regards by simpler alternatives such as ER [14]. Replay methods that use heuristics in sampling [16, 18, 19] are unable to address the drastic domain-shift of a long stream of tasks and result in sudden performance degradation for large domain-shifts as shown in Fig. 4. Likewise, Regularization methods [30, 41] perform similarly to a naive baseline, SGD. Parameter-isolation methods [7, 23] fail to train due to memory requirements.

We analyze the Total Cost ($\mathbb{T}_c$) as the space complexity of the buffer, memory and any auxiliary information specific to the method. We denote by $|x|$ the input dimension, $|\theta|$ the number of model parameters, $|\phi|$ the total size of the intermediate representations ($|\phi_{-2}|$ the penultimate feature size, and $|\phi_{-1}|$ the logit size). iCaRL uses a *herding* buffer strategy that computes artifacts for the entire dataset to subsample, which results in higher than theoretical peak memory usage. For our method, we use an efficient implementation to calculate $\mathcal{L}_{bmc}$ that does not transmit ϕ. As such, BMC maintains a memory footprint per task $O(|x|) + k\theta$.

We compute $\mathbb{T}_c$ in megabytes (MB) for a given buffer size, and thus $\mathbb{A}_c$ represents the improvement in accuracy per unit of MB Eq. (6). Our best performing model variant achieves an accuracy of 71.87% with a Buffer and Memory size of 15k respectively. For our method we find the Pareto optimal configuration with regards to $\mathbb{A}_c$ and use Memory and Buffer size of 15k and 10k respectively, Tab. 1. When comparing between methods, there are limitations in using $\mathbb{A}_c$ for a single configuration. The comparison for the

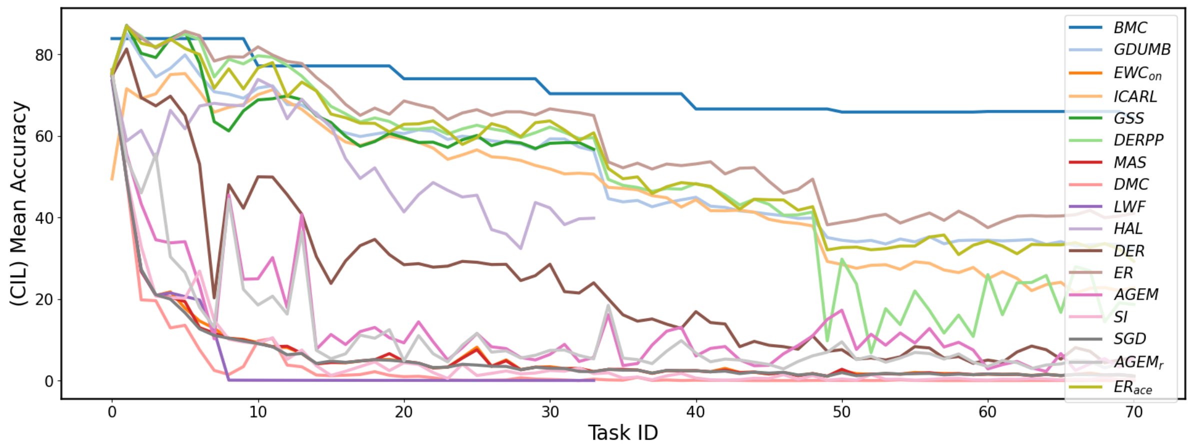

Figure 4. BMC outperforms all other methods on the average accuracy for the Stream dataset and under a CIL evaluation setting.

hyper-parameter range that each method was trained on and might be optimal for $\overline{A}$ but not be Pareto optimal in terms of $\mathbb{A}_c$. To this end, we motivate that the evaluation of $\mathbb{A}_c$ is done for multiple configurations. We provide additional details on the limitation of $\mathbb{A}_c$ in Sec. 6 and provide the Pareto front of our method in the supplementary that can serve as a baseline for future work.

5.2. Ablation Study and Analysis

We conduct an ablation study on the Permuted-MNIST [123] to experimentally verify the impact of each component in BMC. We focus our ablation experiment on different hyper-parameter effects during regularization and consolidation phases and summarize our findings in Fig. 5, where the component settings are compared by the mean accuracy. In total we run 629 experiments and for each experiment, we use a different random seed where we uniformly sample values for each hyper-parameter in the reported range. For continuous hyper-parameters and for reasons of clarity, we report the interpolated curve of all random trials. Lastly, for brevity we discuss the most important components in this section and provide experimental results on additional hyper-parameters in the supplementary.

Number of experts. The average accuracy increases monotonically with a larger batch of experts. Consolidating more experts at each step improves convergence on all tasks and reduces gradient noise. Our results agree with our hypothesis in Sec. 3 and 4, where we claim that batched distillation is a better approximation to the joint optimization goal of all tasks.

Consolidation & Stability loss. We compare the effectiveness of different alternatives for *stability loss* and *batched distillation loss*. We compare our method against Knowledge Distillation (KD) as used by [18, 19]. In both the regularization and consolidation phases, the pairwise intermediate feature distillation $\mathcal{L}_{bd}$ performed better for BMC.

Stability & Consolidation coef. Stability coefficient (λ) scales the stability loss, while the consolidation coefficient (β) scales $\mathcal{L}_{bmc}$ during the consolidation phase. Results show the final performance is positively correlated to λ and β with linear correlation coefficients of 4.12e-2 and 9.39e-2 respectively. Our findings agree with our hypothesis that a stronger batched distillation penalty provides a better approximation to the *multi-task gradient* Sec. 3.

Sampling method. We evaluate more sophisticated sampling strategies, including gradient-based [17] and compare with random selection. We found that random sampling significantly outperformed every other sampling method. We argue that more sophisticated methods in sampling are not suitable for a long sequence of tasks as any inductive bias in selecting which samples to store can be biased either to more recent or later tasks and lead to under-represented tasks or classes. Our results are in agreement with the previous survey [2].

Buffer-Memory size. We vary the size of buffer and memory to study the effect they have on accuracy. As expected, the performance of our method increases with the increased buffer-memory. We find, however, that the effect of buffer size is greater with a linear corr. coefficient of 0.193 when compared to memory with a coef. of 0.104.

6. Discussion

Previous work [18, 119] shows improvement in standardized benchmarks compared to simpler approaches such as ER [14]. We reason that the results hold for a short sequence of similar tasks or synthetically generated datasets. In contrast, we find that for the Stream dataset ER outperforms other baseline methods significantly. Works built on top of ER [18, 19] that use an auxiliary loss suffer in performance for the same benchmark. The reason for the performance degradation has to be studied further, but we hypoth-

Methods	S-CIFAR (10)	S-CIFAR (20)	Tiny (10)	Stream (71)	$\mathbb{A}_c \uparrow$	$\mathbb{T}_c$	Time $\downarrow$
SGD	8.5	3.7	7.9	2.1	-	$O(1)$	100%
Multi-Task	75.79	75.79	68.89	89.3	-	$O(1)$	100%
ER	12.4[3]	14.4[2]	27.4[1]	41.4	5.40	$O(\|x\|)$	184%
DER++	27.0[2]	-	39.0[1]	19.4	0.53	$O(\|x\| + \|\phi_{-1}\|)$	205%
A-GEM	6.5[2]	3.6[2]	8.0[1]	6.6	0.67	$O(\|x\|) + 2\|\theta\|$	231%
iCaRL	25.5[3]	19.2[2]	14.1[1]	23.4	0.59	$O(\|x\| + \|\phi_{-2:}\|) + \|\theta\|$	141%
GDumb	36.0[2]	22.1[2]	-	33.0	4.29	$O(\|x\|)$	129%
GSS	17.4[2]	11.3[2]	-	-	-	$O(\|x\|)$	1203%
DMC	36.2	-	-	1.0	0.02	$O(1) + \|\theta\|$	140%
EWC_{on}	13.1	3.7	7.6	2.1	0.99	$O(1) + 2\|\theta\|$	172%
Ours	66.5[4]	67.4[4]	49.4[4]	**70.4**	**6.27**	$O(\|x\|) + k\theta$	**78%**

Table 1. CIL performance on split CIFAR-100 (S-CIFAR) for 10 and 20 tasks, split Tiny-ImageNet for 10 tasks and Stream Dataset for 71 tasks. Baseline results for S-CIFAR and Tiny-ImageNet are from [2, 18, 119–122] with the buffer size annotated next to the reported result as 5120[1], 5000[2], 1000[3] or 2000[4]. We use '-' to denote results that are not available in the literature, not applicable or not feasible to obtain.

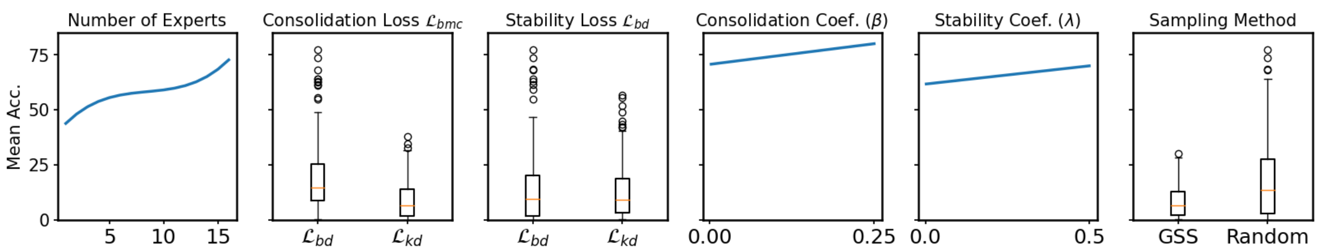

Figure 5. Hyper-parameter ablation results on Permuted-MNIST for 128 tasks and 1280 classes. From left to right, we vary the number of expert models, the loss used in consolidation and *stability loss*, the consolidation and stability coefficients, and finally the sampling method for constructing the buffer and memory.

esize that the auxiliary loss introduces the task-recency bias within the artifacts it is applied to.

Total Cost $\mathbb{T}_c$ and by extension $\mathbb{A}_c$ do not take into consideration of the hyper-parameters used for each method that are flexible design choices and as such can be non-equivalent when comparing two methods and for a single configuration. Consider that the number of parameters for a model can change for a different configuration and can lead to a method with degradation in $\mathbb{A}_c$ if the method stores intermediate gradients [16] or features [19] compared to methods that only store logits [18]. For a thorough evaluation between methods that is agnostic to the hyper-parameter a Pareto optimal configuration must be used to evaluate each method. Additionally, an improved version of $\mathbb{A}_c$ that is not influenced by component hyper-parameter choice can provide a better evaluation metric.

7. Conclusion

In this paper, we propose Batch Model Consolidation, a Continual Learning framework that reduces catastrophic forgetting when training on a long sequence of tasks from diverse domains and ranging difficulties. Our method combines Regularization, with the use of the stability loss on a previously trained base model; Replay, with the use of batched distillation loss on memory data; and Parameter-Isolation where multiple expert models are trained on a sequence of disjoint tasks. Lastly, we extend our framework to work in a distributed setting where each expert can reside on a different device and specialize in a given task. We experimentally demonstrate that BMC is the only method that maintains performance for our long sequence of 71 tasks. Lastly, we make the code of this work publicly available so that it can serve as a benchmark for future work in Distributed Continual Learning.

8. Acknowledgement

This work was supported by C-BRIC (one of six centers in JUMP, a Semiconductor Research Corporation (SRC) program sponsored by DARPA), DARPA (HR00112190134) and the Army Research Office (W911NF2020053). The authors affirm that the views expressed herein are solely their own, and do not represent the views of the United States government or any agency thereof.

References

[1] Raia Hadsell, Dushyant Rao, Andrei A. Rusu, and Razvan Pascanu. Embracing change: Continual learning in deep neural networks. *Trends in Cognitive Sciences*, 24(12):1028–1040, 2020. 1

[2] Marc Masana, Xialei Liu, Bartlomiej Twardowski, Mikel Menta, Andrew D. Bagdanov, and Joost van de Weijer. Class-incremental learning: survey and performance evaluation. *CoRR*, abs/2010.15277, 2020. 1, 3, 4, 6, 7, 8

[3] 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. 1

[4] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009. 1, 6

[5] Ya Le and Xuan Yang. Tiny imagenet visual recognition challenge. *CS 231N*, 7(7):3, 2015. 1, 6

[6] Matthias De Lange, Rahaf Aljundi, Marc Masana, Sarah Parisot, Xu Jia, Ales Leonardis, Gregory G. Slabaugh, and Tinne Tuytelaars. Continual learning: A comparative study on how to defy forgetting in classification tasks. *CoRR*, abs/1909.08383, 2019. 1, 3, 6

[7] Andrei A. Rusu, Neil C. Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. Progressive neural networks. *CoRR*, abs/1606.04671, 2016. 1, 2, 3, 6

[8] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13):3521–3526, mar 2017. 1, 3

[9] Heechul Jung, Jeongwoo Ju, Minju Jung, and Junmo Kim. Less-forgetting learning in deep neural networks. *CoRR*, abs/1607.00122, 2016. 1, 3

[10] Sang-Woo Lee, Jin-Hwa Kim, JungWoo Ha, and Byoung-Tak Zhang. Overcoming catastrophic forgetting by incremental moment matching. *CoRR*, abs/1703.08475, 2017. 1

[11] Friedemann Zenke, Ben Poole, and Surya Ganguli. Continual learning through synaptic intelligence. In *International Conference on Machine Learning*, pages 3987–3995. PMLR, 2017. 1, 3, 4, 6

[12] Chen Zhang, Yu Xie, Hang Bai, Bin Yu, Weihong Li, and Yuan Gao. A survey on federated learning. *Knowledge-Based Systems*, 216:106775, 2021. 1

[13] Matthias De Lange, Rahaf Aljundi, Marc Masana, Sarah Parisot, Xu Jia, Aleš Leonardis, Gregory Slabaugh, and Tinne Tuytelaars. A continual learning survey: Defying forgetting in classification tasks. *IEEE transactions on pattern analysis and machine intelligence*, 44(7):3366–3385, 2021. 2

[14] Matthew Riemer, Ignacio Cases, Robert Ajemian, Miao Liu, Irina Rish, Yuhai Tu, and Gerald Tesauro. Learning to learn without forgetting by maximizing transfer and minimizing interference. *arXiv preprint arXiv:1810.11910*, 2018. 2, 6, 7

[15] David Lopez-Paz and Marc'Aurelio Ranzato. Gradient episodic memory for continuum learning. *CoRR*, abs/1706.08840, 2017. 2

[16] Arslan Chaudhry, Marc'Aurelio Ranzato, Marcus Rohrbach, and Mohamed Elhoseiny. Efficient lifelong learning with a-gem. *arXiv preprint arXiv:1812.00420*, 2018. 2, 6, 8

[17] Rahaf Aljundi, Min Lin, Baptiste Goujaud, and Yoshua Bengio. Gradient based sample selection for online continual learning. *Advances in neural information processing systems*, 32, 2019. 2, 4, 6, 7

[18] Pietro Buzzega, Matteo Boschini, Angelo Porrello, Davide Abati, and SIMONE CALDERARA. Dark experience for general continual learning: a strong, simple baseline. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 15920–15930. Curran Associates, Inc., 2020. 2, 3, 6, 7, 8

[19] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, and Christoph H. Lampert. icarl: Incremental classifier and representation learning. *CoRR*, abs/1611.07725, 2016. 2, 3, 6, 7, 8

[20] Ari S Benjamin, David Rolnick, and Konrad Kording. Measuring and regularizing networks in function space. *arXiv preprint arXiv:1805.08289*, 2018. 2

[21] Junting Zhang, Jie Zhang, Shalini Ghosh, Dawei Li, Serafettin Tasci, Larry P. Heck, Heming Zhang, and C.-C. Jay Kuo. Class-incremental learning via deep model consolidation. *CoRR*, abs/1903.07864, 2019. 2, 3, 4, 6

[22] Antonio Carta, Andrea Cossu, Vincenzo Lomonaco, and Davide Bacciu. Ex-model: Continual learning from a stream of trained models. *CoRR*, abs/2112.06511, 2021. 2, 3, 4

[23] Mitchell Wortsman, Vivek Ramanujan, Rosanne Liu, Aniruddha Kembhavi, Mohammad Rastegari, Jason Yosinski, and Ali Farhadi. Supermasks in superposition. *CoRR*, abs/2006.14769, 2020. 2, 6

[24] Joan Serra, Didac Suris, Marius Miron, and Alexandros Karatzoglou. Overcoming catastrophic forgetting with hard attention to the task. In *International Conference on Machine Learning*, pages 4548–4557. PMLR, 2018. 2

[25] Brian Cheung, Alexander Terekhov, Yubei Chen, Pulkit Agrawal, and Bruno Olshausen. Superposition of many models into one. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. 2

[26] Yujun Shi, Li Yuan, Yunpeng Chen, and Jiashi Feng. Continual learning via bit-level information preserving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16674–16683, 2021. 2

[27] Rahul Ramesh and Pratik Chaudhari. Boosting a model zoo for multi-task and continual learning. *CoRR*, abs/2106.03027, 2021. 3

[28] Arun Mallya and Svetlana Lazebnik. Packnet: Adding multiple tasks to a single network by iterative pruning. In

2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 7765–7773, 2018. 3

[29] Prakhar Kaushik, Alex Gain, Adam Kortylewski, and Alan L. Yuille. Understanding catastrophic forgetting and remembering in continual learning with optimal relevance mapping. *CoRR*, abs/2102.11343, 2021. 3

[30] 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)*, pages 139–154, 2018. 3, 4, 6

[31] Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE transactions on pattern analysis and machine intelligence*, 40(12):2935–2947, 2017. 3, 4

[32] Yuhang Ma, Zhongle Xie, Jue Wang, Ke Chen, and Lidan Shou. Continual federated learning based on knowledge distillation. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence*, July 2022. 3

[33] Jaehong Yoon, Wonyong Jeong, Giwoong Lee, Eunho Yang, and Sung Ju Hwang. Federated continual learning with adaptive parameter communication. *CoRR*, abs/2003.03196, 2020. 3

[34] Geoffrey Hinton, Oriol Vinyals, Jeff Dean, et al. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2(7), 2015. 3

[35] Mingi Ji, Byeongho Heo, and Sungrae Park. Show, attend and distill: Knowledge distillation via attention-based feature matching. *CoRR*, abs/2102.02973, 2021. 3

[36] Byeongho Heo, Jeesoo Kim, Sangdoo Yun, Hyojin Park, Nojun Kwak, and Jin Young Choi. A comprehensive overhaul of feature distillation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019. 3

[37] Stanislaw Jastrzebski, Zachary Kenton, Devansh Arpit, Nicolas Ballas, Asja Fischer, Yoshua Bengio, and Amos J. Storkey. Three factors influencing minima in SGD. *CoRR*, abs/1711.04623, 2017. 4

[38] Sam McCandlish, Jared Kaplan, Dario Amodei, and OpenAI Dota Team. An empirical model of large-batch training. *CoRR*, abs/1812.06162, 2018. 4

[39] Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. On large-batch training for deep learning: Generalization gap and sharp minima. *CoRR*, abs/1609.04836, 2016. 4

[40] Elad Hoffer, Itay Hubara, and Daniel Soudry. Train longer, generalize better: closing the generalization gap in large batch training of neural networks. *Advances in neural information processing systems*, 30, 2017. 4

[41] Jonathan Schwarz, Wojciech Czarnecki, Jelena Luketina, Agnieszka Grabska-Barwinska, Yee Whye Teh, Razvan Pascanu, and Raia Hadsell. Progress & compress: A scalable framework for continual learning. In *International Conference on Machine Learning*, pages 4528–4537. PMLR, 2018. 4, 6

[42] Yan Hong, Li Niu, Jianfu Zhang, and Liqing Zhang. Beyond without forgetting: Multi-task learning for classification with disjoint datasets. In *2020 IEEE International Conference on Multimedia and Expo (ICME)*, pages 1–6, 2020. 5

[43] Subhransu Maji, Esa Rahtu, Juho Kannala, Matthew B. Blaschko, and Andrea Vedaldi. Fine-grained visual classification of aircraft. *CoRR*, abs/1306.5151, 2013. 6

[44] Aleksandr Antonov. Apparel images dataset. https://www.kaggle.com/datasets/trolukovich/apparel-images-dataset. Accessed: 2022-10-30. 6

[45] Asia Pacific Tele-Ophthalmology Society (APTOS). APTOS 2019 blindness detection. https://www.kaggle.com/competitions/aptos2019-blindness-detection/overview. Accessed: 2022-10-30. 6

[46] Wei Ren Tan, Chee Seng Chan, Hernan Aguirre, and Kiyoshi Tanaka. Improved artgan for conditional synthesis of natural image and artwork. *IEEE Transactions on Image Processing*, 28(1):394–409, 2019. 6

[47] Akash Nagaraj. ASL Alphabet: Image data set for alphabets in the American Sign Language. https://www.kaggle.com/datasets/grassknoted/asl-alphabet. Accessed: 2022-10-30. 6

[48] Pierre-Alexandre Clorichel. Boat types recognition: About 1,500 pictures of boats classified in 9 categories. https://www.kaggle.com/datasets/clorichel/boat-types-recognition. Accessed: 2022-11-10. 6

[49] Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization. In *4th International IEEE Workshop on 3D Representation and Recognition (3dRR-13)*, Sydney, Australia, 2013. 6

[50] jr2ngb (username). Cataract dataset. https://www.kaggle.com/datasets/jr2ngb/cataractdataset. Accessed: 2022-11-10. 6

[51] Shuo Yang, Ping Luo, Chen Change Loy, and Xiaoou Tang. From facial parts responses to face detection: A deep learning approach. *CoRR*, abs/1509.06451, 2015. 6

[52] Jakob Nikolas Kather, Cleo-Aron Weis, Francesco Bianconi, Susanne M Melchers, Lothar R Schad, Timo Gaiser, Alexander Marx, and Frank Gerrit Zöllner. Multi-class texture analysis in colorectal cancer histology. *Scientific reports*, 6(1):1–11, 2016. 6

[53] Ç. F. Özgenel. Concrete crack images for classification. *Mendeley Data*, V2, 2019. 6

[54] Vincenzo Lomonaco and Davide Maltoni. Core50: a new dataset and benchmark for continuous object recognition. In *Conference on Robot Learning*, pages 17–26. PMLR, 2017. 6

[55] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. The Caltech-UCSD Birds-200-2011 dataset. Technical Report CNS-TR-2011-001, California Institute of Technology, 2011. 6

[56] Alex Olsen, Dmitry A Konovalov, Bronson Philippa, Peter Ridd, Jake C Wood, Jamie Johns, Wesley Banks, Benjamin Girgenti, Owen Kenny, James Whinney, et al. Deepweeds: A multiclass weed species image dataset for deep learning. *Scientific reports*, 9(1):1–12, 2019. 6

[57] Shubham Goel and Bill Hall. Dermnet: Image data for 23 categories of skin diseases. https://www.kaggle.com/datasets/shubhamgoel27/dermnet. Accessed: 2022-11-10. 6

[58] M. Cimpoi, S. Maji, I. Kokkinos, S. Mohamed, , and A. Vedaldi. Describing textures in the wild. In *Proceedings of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2014. 6

[59] Sunim Acharya. Electronic components and devices: Dataset containing major electrical and electronic components and devices. https://www.kaggle.com/datasets/aryaminus/electronic-components. Accessed: 2022-11-10. 6

[60] Gregory Cohen, Saeed Afshar, Jonathan Tapson, and André van Schaik. EMNIST: an extension of MNIST to handwritten letters. *CoRR*, abs/1702.05373, 2017. 6

[61] Patrick Helber, Benjamin Bischke, Andreas Dengel, and Damian Borth. Eurosat: A novel dataset and deep learning benchmark for land use and land cover classification. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 2019. 6

[62] Li-Jia Li and Li Fei-Fei. What, where and who? classifying events by scene and object recognition. In *2007 IEEE 11th international conference on computer vision*, pages 1–8. IEEE, 2007. 6

[63] Shiekh Burhan. Face mask dataset: Covid-19 dataset for training face mask classifier. https://www.kaggle.com/datasets/shiekhburhan/face-mask-dataset. Accessed: 2022-11-10. 6

[64] Param Aggarwal. Fashion product images dataset: 44k products with multiple category labels, descriptions and high-res images. https://www.kaggle.com/datasets/paramaggarwal/fashion-product-images-dataset. Accessed: 2022-11-10. 6

[65] Manas Sambare. Fer-2013: Learn facial expressions from an image. https://www.kaggle.com/datasets/msambare/fer2013. Accessed: 2022-11-10. 6

[66] Parneet Kaur, , Karan Sikka, Weijun Wang, serge Belongie, and Ajay Divakaran. Foodx-251: A dataset for fine-grained food classification. *arXiv preprint arXiv:1907.06167*, 2019. 6

[67] Oguzhan Ulucan, Diclehan Karakaya, and Mehmet Turkan. A large-scale dataset for fish segmentation and classification. In *2020 Innovations in Intelligent Systems and Applications Conference (ASYU)*, pages 1–5. IEEE, 2020. 6

[68] M.-E. Nilsback and A. Zisserman. A visual vocabulary for flower classification. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 1447–1454, 2006. 6

[69] Lukas Bossard, Matthieu Guillaumin, and Luc Van Gool. Food-101 – mining discriminative components with random forests. In *European Conference on Computer Vision*, 2014. 6

[70] Philipp Jund, Nichola Abdo, Andreas Eitel, and Wolfram Burgard. The freiburg groceries dataset. *CoRR*, abs/1611.05799, 2016. 6

[71] Henry W Leung and Jo Bovy. Deep learning of multi-element abundances from high-resolution spectroscopic data. *Monthly Notices of the Royal Astronomical Society*, nov 2018. 6

[72] Mostafa Mohamed. Garbage classification (12 classes): Images dataset for classifying household garbage. https://www.kaggle.com/datasets/mostafaabla/garbage-classification. Accessed: 2022-11-10. 6

[73] J. Stallkamp, M. Schlipsing, J. Salmen, and C. Igel. Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition. *Neural Networks*, 32:323–332, 2012. Selected Papers from IJCNN 2011. 6

[74] Philipp Tschandl, Cliff Rosendahl, and Harald Kittler. The ham10000 dataset, a large collection of multi-source dermatoscopic images of common pigmented skin lesions. *Scientific data*, 5(1):1–9, 2018. 6

[75] Olga Belitskaya. Classification of handwritten letters: Images of russian letters. https://www.kaggle.com/datasets/olgabelitskaya/classification-of-handwritten-letters. Accessed: 2022-11-10. 6

[76] Rémi Ratajczak, Carlos F Crispim-Junior, Élodie Faure, Béatrice Fervers, and Laure Tougne. Automatic Land Cover Reconstruction From Historical Aerial Images: An Evaluation of Features Extraction and Classification Algorithms. *IEEE Transactions on Image Processing*, Jan. 2019. 6

[77] Visipedia. inaturalist 2021 competition: Fgvc8 workshop at cvpr. https://github.com/visipedia/inat_comp/tree/master/2021. Accessed: 2022-10-30. 6

[78] Ariadna Quattoni and Antonio Torralba. Recognizing indoor scenes. In *2009 IEEE conference on computer vision and pattern recognition*, pages 413–420. IEEE, 2009. 6

[79] Puneet Bansal. Intel image classification: Image scene classification of multiclass. https://www.kaggle.com/datasets/puneet6060/intel-image-classification. Accessed: 2022-11-10. 6

[80] Xiaoping Wu, Chi Zhan, Yu-Kun Lai, Ming-Ming Cheng, and Jufeng Yang. Ip102: A large-scale benchmark dataset for insect pest recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 6

[81] Daniel S Kermany, Michael Goldbaum, Wenjia Cai, Carolina CS Valentim, Huiying Liang, Sally L Baxter, Alex McKeown, Ge Yang, Xiaokang Wu, Fangbing Yan, et al. Identifying medical diagnoses and treatable diseases by image-based deep learning. *Cell*, 172(5):1122–1131, 2018. 6

[82] Manish KC (username). The kvasir-capsule dataset. https://www.kaggle.com/datasets/manishkc06/the-kvasircapsule-dataset. Accessed: 2022-11-10. 6

[83] Yi Yang and Shawn Newsam. Bag-of-visual-words and spatial extensions for land-use classification. In *Proceedings of the 18th SIGSPATIAL international conference on advances in geographic information systems*, pages 270–279, 2010. 6

[84] Joost Hazelzet. Images of lego bricks: 40,000 images of 50 different lego bricks. https://www.kaggle.

com/datasets/joosthazelzet/lego-brick-images. Accessed: 2022-11-10. 6

[85] Joey Lim Zy. Historical building (malacca, malaysia): 162 images of historical buildings in malaysia. https://www.kaggle.com/datasets/joeylimzy/historical-building-malacca-malaysia. Accessed: 2022-11-10. 6

[86] Mert Koklu. Manga facial expressions: Facial expressions of manga (japanese comic) character faces. https://www.kaggle.com/datasets/mertkkl/manga-facial-expressions. Accessed: 2022-11-10. 6

[87] YoucefATTALLAH97 (username). Minerals identification & classification: Minet v2. https://www.kaggle.com/datasets/youcefattallah97/minerals-identification-classification. Accessed: 2022-11-10. 6

[88] Hemanth Venkateswara, Jose Eusebio, Shayok Chakraborty, and Sethuraman Panchanathan. Deep hashing network for unsupervised domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5018–5027, 2017. 6

[89] Daniel Ma, Gerald Friedland, and Mario Michael Krell. Origamiset1. 0: Two new datasets for origami classification and difficulty estimation. *arXiv preprint arXiv:2101.05470*, 2021. 6

[90] James Philbin, Ondrej Chum, Michael Isard, Josef Sivic, and Andrew Zisserman. Object retrieval with large vocabularies and fast spatial matching. In *2007 IEEE conference on computer vision and pattern recognition*, pages 1–8. IEEE, 2007. 6

[91] Bastiaan S. Veeling, Jasper Linmans, Jim Winkens, Taco Cohen, and Max Welling. Rotation equivariant CNNs for digital pathology. In Alejandro F. Frangi, Julia A. Schnabel, Christos Davatzikos, Carlos Alberola-López, and Gabor Fichtinger, editors, *Medical Image Computing and Computer Assisted Intervention – MICCAI 2018*, pages 210–218, Cham, 2018. Springer International Publishing. 6

[92] 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. 6

[93] Emirhan BULUT. Planets and moons dataset - ai in space: A public dataset for large-scale multi-label and multi-class image classification. *Dataset available from https://github.com/emirhanai/Planets-and-Moons-Dataset-AI-in-Space and https://www.kaggle.com/datasets/emirhanai/planets-and-moons-dataset-ai-in-space*, 2022. 6

[94] Davinder Singh, Naman Jain, Pranjali Jain, Pratik Kayal, Sudhakar Kumawat, and Nipun Batra. Plantdoc: A dataset for visual plant disease detection. In *Proceedings of the 7th ACM IKDD CoDS and 25th COMAD*, CoDS COMAD 2020, page 249–253, New York, NY, USA, 2020. Association for Computing Machinery. 6

[95] Paul Mooney. Chest x-ray images (pneumonia). https://www.kaggle.com/datasets/paultimothymooney/chest-xray-pneumonia. Accessed: 2022-11-10. 6

[96] Lance Zhang. 7000 hand-cropped and labeled Pokemon images for classification. https://www.kaggle.com/datasets/lantian773030/pokemonclassification. Accessed: 2022-11-10. 6

[97] Hyun Oh Song, Yu Xiang, Stefanie Jegelka, and Silvio Savarese. Deep metric learning via lifted structured feature embedding. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4004–4012, 2016. 6

[98] Gong Cheng, Junwei Han, and Xiaoqiang Lu. Remote sensing image scene classification: Benchmark and state of the art. *Proceedings of the IEEE*, 105(10):1865–1883, Oct 2017. 6

[99] Murat Koklu, Ilkay Cinar, and Yavuz Selim Taspinar. Classification of rice varieties with deep learning methods. *Computers and electronics in agriculture*, 187:106285, 2021. 6

[100] Shahriar Hossain, Jahir Uddin, and Rakibul Alam Nahin. Rock classification dataset: Multi class classification using different types of images of rocks. https://www.kaggle.com/datasets/salmaneunus/rock-classification. Accessed: 2022-11-10. 6

[101] RobinReni (username). House rooms image dataset. https://www.kaggle.com/datasets/robinreni/house-rooms-image-dataset. Accessed: 2022-11-10. 6

[102] Adam W Harley, Alex Ufkes, and Konstantinos G Derpanis. Evaluation of deep convolutional nets for document image classification and retrieval. In *International Conference on Document Analysis and Recognition (ICDAR)*, 2015. 6

[103] Deep Contractor (username). IS THAT SANTA? (image classification): Santa Claus classification. https://www.kaggle.com/datasets/deepcontractor/is-that-santa-image-classification. Accessed: 2022-11-10. 6

[104] san_bt (username). Satellite images to predict poverty: Images taken over the region of africa for research purpose. https://www.kaggle.com/datasets/sandeshbhat/satellite-images-to-predict-povertyafrica. Accessed: 2022-11-10. 6

[105] Alexandre Attia. Simpson recognition. https://github.com/alexattia/SimpsonRecognition. Accessed: 2022-11-10. 6

[106] Mathias Eitz, James Hays, and Marc Alexa. How do humans sketch objects? *ACM Trans. Graph. (Proc. SIGGRAPH)*, 31(4):44:1–44:10, 2012. 6

[107] Gerry (username). 100 sports image classification. https://www.kaggle.com/datasets/gpiosenka/sports-classification. Accessed: 2022-11-10. 6

[108] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y. Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS Workshop on Deep Learning and Unsupervised Feature Learning 2011*, 2011. 6

[109] Yibin Huang. Textures classification dataset. https://github.com/abin24/Textures-Dataset. Accessed: 2022-11-10. 6
[110] M Israk Ahmed, Shahriyar Mahmud Mamun, and Asif Uz Zaman Asif. Dcnn-based vegetable image classification using transfer learning: A comparative study. In *2021 5th International Conference on Computer, Communication and Signal Processing (ICCCSP)*, pages 235–243. IEEE, 2021. 6
[111] Felice Pollano. Watermark dataset builder. https://github.com/FelicePollano/WatermarkDataSetBuilder. Accessed: 2022-11-10. 6
[112] A Gbeminiyi. Multi-class weather dataset for image classification. *Mendeley Data*, 2018. 6
[113] Dominic Monn. Clothing & models: A collection of clothing pieces, scraped from zalando.com. https://www.kaggle.com/datasets/dqmonn/zalando-store-crawl. Accessed: 2022-11-10. 6
[114] Kaggle. Kaggle: Your Home for Data Science. https://www.kaggle.com. Accessed: 2022-11-10. 6
[115] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. *CoRR*, abs/2103.00020, 2021. 6
[116] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015. 6
[117] Ameya Prabhu, Philip HS Torr, and Puneet K Dokania. Gdumb: A simple approach that questions our progress in continual learning. In *European conference on computer vision*, pages 524–540. Springer, 2020. 6
[118] Arslan Chaudhry, Albert Gordo, Puneet Dokania, Philip Torr, and David Lopez-Paz. Using hindsight to anchor past knowledge in continual learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 6993–7001, 2021. 6
[119] Zheda Mai, Ruiwen Li, Jihwan Jeong, David Quispe, Hyunwoo Kim, and Scott Sanner. Online continual learning in image classification: An empirical survey. *Neurocomputing*, 469:28–51, 2022. 7, 8
[120] Yiduo Guo, Bing Liu, and Dongyan Zhao. Online continual learning through mutual information maximization. In *International Conference on Machine Learning*, pages 8109–8126. PMLR, 2022. 8
[121] Pietro Buzzega, Matteo Boschini, Angelo Porrello, and Simone Calderara. Rethinking experience replay: a bag of tricks for continual learning. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pages 2180–2187. IEEE, 2021. 8
[122] Runkai Zheng, Zhijia Yu, Yinqi Zhang, Chris Ding, Hei Victor Cheng, and Li Liu. Learning class unique features in fine-grained visual classification. *arXiv preprint arXiv:2011.10951*, 2020. 8
[123] Yann LeCun. The mnist database of handwritten digits. *http://yann. lecun. com/exdb/mnist/*, 1998. 7