# 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

## A. Experiment Details

### A.1. Stream Dataset Details

In total, we pre-process and use 71 datasets from the computer vision literature and Kaggle [1–71]. We use the train and validation split from each original dataset and for each task. Some datasets have multiple ways to label an image, for example, CelebA [9] assigns 40 binary labels to each image, such as ‘Brown Hair’ or ‘Blurry’. We use ‘Sub Task’ to denote which split or sub-task we use for each dataset and refer the reader to the original dataset documentation on the details of each split. We do not modify or alter any of the datasets in any way. To speed up training and evaluation, we extract the image feature vectors for each dataset using a CLIP model and use them for the classification task directly. Tab. 6 and Tab. 7 contain the statistics for each dataset.

**Task difficulty**. SGD performance can be used as a proxy for task difficulty [72]. Some tasks, such as Planets [51] (task id 50), have really low difficulty and thus high performance on SGD. The inflated performance on the task can be caused by over-fitting or be a reflection of the poor curation process and similarity between the train and validation split of the dataset. The purpose of the Stream dataset is to introduce nuances between tasks, such as with the varying training dataset size. As such, we welcome the nuances and errors that are inherited in each task, as they are a reflection of a realistic training and evaluation scenario. Since all methods are evaluated under the same conditions, the comparison is equivalent. Lastly, the Stream dataset contains a diversity of images, reflected by the curation protocol used to compose each task dataset.

### A.2. Training Configurations

We report the performance of SGD on each task in Tab. 6 and Tab. 7. SGD performance can be considered as an upper bound for our model and train configuration. We compute SGD performance by training on the task in an isolated manner and without applying any method to mitigate forgetting. We evaluate SGD performance on the validation set of each task.

When evaluating the time performance of each method we compute the total time for the method to train on a task. The total duration of the training run can include a warm-up and post-train phase as part of each method. We use CIFAR-100 as an auxiliary dataset for DMC [73].

The train configuration is reported in Tab. 1. We reset the learning rate before and after each rehearsal episode. For both the baseline methods and our method, we use an MLP with residual connections as a backbone model. We apply Batch Normalization [74] after every layer and the ReLU activation function [75]. All experiments run on identical hardware of a V100 GPU cluster and we distribute the workload using Ray [76].

We provide in Tab. 2 and Tab. 3 the hyper-parameters used in Stream Benchmark experiments. We use the reported hyper-parameters for all baseline methods. When experimenting BMC on split CIFAR-100 and split Tiny-ImageNet, we use a memory size of 2,000 and a buffer size of 200 with all other settings the same as learning on Stream Benchmark.

| **Benchmark** | Num. tasks |
|---|---|
| Optimizer | SGD [77] |
| Rehearsal Scheduler | ReduceLROnPlateau[1] |
| Learning Rate | 0.1 |
| Train epochs per Task | 2 |
| Rehearsal epochs per Task | 100 |
| GPU | V100 |
| Model | Residual MLP |
| Res. Blocks | 2 |
| Res. Layers | 3 |
| Dropout | 0.3 |
| Res. Dim | 256 |
| Hidden Dim | 128 |
| Initialization | Xavier [78] |

Table 1. Train Configuration. Detailed overview in Appendix A.2.

[1] https://pytorch.org/docs/stable/generated/torch.optim.lr_scheduler.ReduceLROnPlateau.htm

| **Baselines** | Hyper-params | Values |
|---|---|---|
| | Memory size | 10,000 |
| ER | Replay coef. | 1.0 |
| DER | Distill coef. | 0.5 |
| DER++ | Replay coef.<br>Distill coef. | 1.0<br>0.5 |
| GDumb | Max/min LR<br>Epochs<br>Cutmix | 5e-2 / 5e-4<br>256<br>None |
| HAL | Penalty coef.<br>Beta<br>Gamma | 0.1<br>0.5<br>0.1 |
| LwF | Penalty coef.<br>Temperature | 0.5<br>2.0 |
| GSS | Minibatch size<br>Batch Num | 10<br>1 |
| DMC | Consolidation LR<br>Consolidation Epochs | 0.05<br>10 |
| EWC$_{on}$ | Penalty coef.<br>Gamma | 0.7<br>1.0 |
| MAS | Penalty coef.<br>Gamma | 0.7<br>1.0 |
| SI | c<br>xi | 0.5<br>1.0 |

Table 2. Hyper-parameter settings for training the baseline methods on Stream Benchmark. Baselines that do not have extra hyper-parameters are omitted (A-GEM, iCaRL).

| **Phase** | Hyper-params | Values |
|---|---|---|
| Regularization | Stability coef.<br>Buffer size | 1.0<br>10,000 |
| Consolidation | Num. Experts<br>Task loss coef.<br>Consolidation coef.<br>Buffer sampling | 10<br>1.0<br>1.0<br>Random |

Table 3. Hyper-parameter settings for training BMC on Stream Benchmark.

## B. Stream Benchmark Analysis

| **Methods** | Stream Benchmark ↑ | Time ↓ |
|---|---|---|
| SGD | 2.1 | 100% |
| Multi-Task | 89.3 | 100% |
| AGEM [79] | 6.6 | 231% |
| AGEM$_R$ [80] | 4.3 | 230% |
| DER [81] | 5.6 | 170% |
| DER++ [81] | 19.4 | 205% |
| DMC [73] | 1.0 | 140% |
| ER [82] | 41.4 | 184% |
| ER-ACE [82] | 29.3 | 184% |
| EWC$_{on}$ [83] | 2.1 | 172% |
| GDumb [84] | 33.0 | 129% |
| GSS [85] | - | 1203%[2] |
| HAL [86] | - | 730%[2] |
| iCaRL [87] | 23.4 | 141% |
| LwF [88] | - | 201%[2] |
| MAS [89] | 2.1 | 144% |
| SI [90] | 1.4 | 515% |
| **BMC** (Ours) | **70.4** | **78%** |

Table 4. CIL performance on Stream Dataset for 71 tasks. We use '-' to denote results that were not feasible to obtain for all 71 tasks due to intractable runtime. [2]Signifies incomplete time-performance evaluation. Detailed explanation of the results in Appendix B

We evaluate a set of recent (*e.g.* DER++ [81]) and old baselines (*e.g.* ER [82]) applicable to our setting, while some other recent baselines have a limited setting, i.e., Transformer models [91]. Recent methods achieved better performance on standard benchmarks (i.e. CIFAR-100) did not outperform a naive baseline (ER) on Stream.

Some methods such as GSS [85] can have an intractable run-time that grows with the number of tasks learned. Other methods, such as LwF [88], have a warm-up stage that requires using the train dataset. Such methods fail to complete past Task id 34 (iNaturalist [35]) since the step of the method is coupled with the size of the task. iNaturalist [35] is made up of 2686843 images, so if a step of a method requires constructing a buffer with new artifacts for each sample [88] or performing multiple back-props [85], the method may not complete the task. The time-performance factor in Tab. 4 may not reflect the failure, such as for LwF [88], since the factor is calculated at the end of training on a task.

We allow all baselines to run uninterrupted for 4 days and we terminate the experiment on the last day. All methods for which we report results finish the benchmark within 24 hours.

### B.1. Normalized Performance

Fig. 1 presents the normalized performance by task difficulty as discussed in Appendix A.1. It can be observed that there is noise when normalizing for task difficulty, or that the method performs better than the upper bound for that task. The results can be explained as an artifact of forward transfer [72] as well as inherited problems with the task dataset, such as overfitting or an imbalanced train-validation split [16].

### B.2. First Task Performance

When evaluating the performance of a method on first task, we observe similar results to the mean task accuracy (Fig. 2), with some exceptions. BMC, our method retains performance on the first task as compared to other baselines. Additionally, the performance difference between mean accuracy and first task accuracy between our method and the baselines is more prominent when evaluating only on the first task. GSS [85], retains performance on the first task but does not learn new tasks. iCaRL [87] outperforms other baselines on the first task accuracy, but it does not perform as well when evaluated in mean accuracy. We hypothesize that this is due to the herding strategy used by iCaRL to compose the buffer that can avoid the task-recency bias.

### B.3. Time Performance Evaluation

Most methods perform similarly in terms of run-time on the benchmark. Some notable exceptions are SI [90], GSS [85] and HAL [86]. Other methods can have a run-time performance that can be seen as non-equivalent. Such methods can perform a step that is agnostic to the task dataset size, such as constructing an auxiliary dataset [73] or training on an auxiliary buffer [84, 87]. As such, the run-time performance of such methods can fluctuate greatly between tasks Fig. 3 (Middle). Additionally, the relative time performance Tab. 4 can appear inflated.

Our method, BMC, can take advantage of training multiple tasks in a distributed fashion and, as such, perform better than the baseline SGD Fig. 3 (**Middle**). Our method has a run-time bottleneck by the largest task in each train incremental step. This is due to the wait operation between each task-batch in order to apply batched distillation loss. For the last task of the 71 datasets, our method is slower than SGD as the performance effect of batched-task incremental learning is not utilized.

### B.4. Pareto Front

The Pareto front of our method shows a trade-off between the Total Cost of a memory and a buffer with Mean Accuracy. We run multiple experiments and vary the buffer size and memory size. We independently sample the Buffer and Memory size configuration between 8k to 20k exemplars at the start of each experiment. The Total Cost represents the mean number of exemplars stored at each train incremental step of our method. The performance gain plateaus as we increase $\mathbb{TC}$. Some configurations are not Pareto optimal, such as the use of a very small buffer and really large memory or vice versa. Both the buffer and memory are integral parts of our method's performance but observe a limitation on the improvement they provide to the performance beyond a certain point. As such we hypothesize that improvements in both the utilization and construction of the buffer and memory are more significant than the size of the buffer.

Lastly, we motivate a method not to be evaluated at a single point when evaluating the cost performance of the method. As can be observed, the relationship between $\mathbb{TC}$ and Mean Accuracy is not linear and requires that more than one configuration be evaluated.

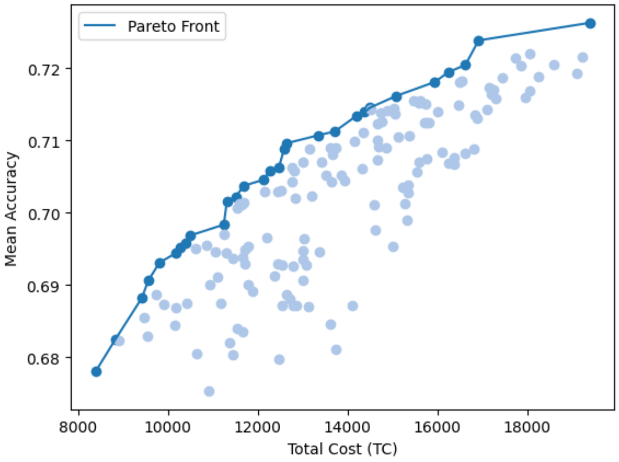


Figure 4. Trade-off between $\mathbb{TC}$ and Mean Accuracy for our method (BMC) with details discussed in Appendix B.4.

## C. Ablation Experiments

We motivate our ablation study in performing an unbiased estimate of each component of our method. Using the benchmark dataset can be a biased estimate of a component of a method for the following reasons. First, we need a dataset for which we have access to a large number of tasks that are evenly divisible by the number of experts we evaluate, i.e. 128 by 16. Second, for the benchmark the tasks have different domain-gaps, for example the difference between 'Lego' and 'Rooms' datasets. Third, the tasks have varying lengths and numbers of classes, such that 'Lego' has 32,000 train images with 46 classes and 'Rooms' has 3,937 images with 5 classes. An ablation study on a dataset with multiple sources of experimental variance requires additional experimental trials for an unbiased estimate. Permuted-MNIST meets all of the above require-

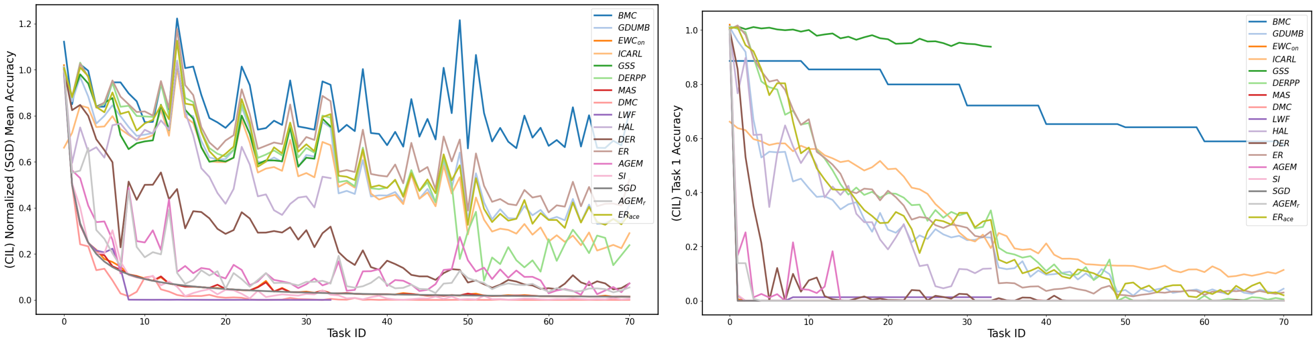


Figure 1. **Left** mean accuracy normalized for task difficulty. Easier tasks are less difficult to forget. **Right** performance on the first task. GSS maintains performance on the first task but fails to learn new tasks Fig. 2.

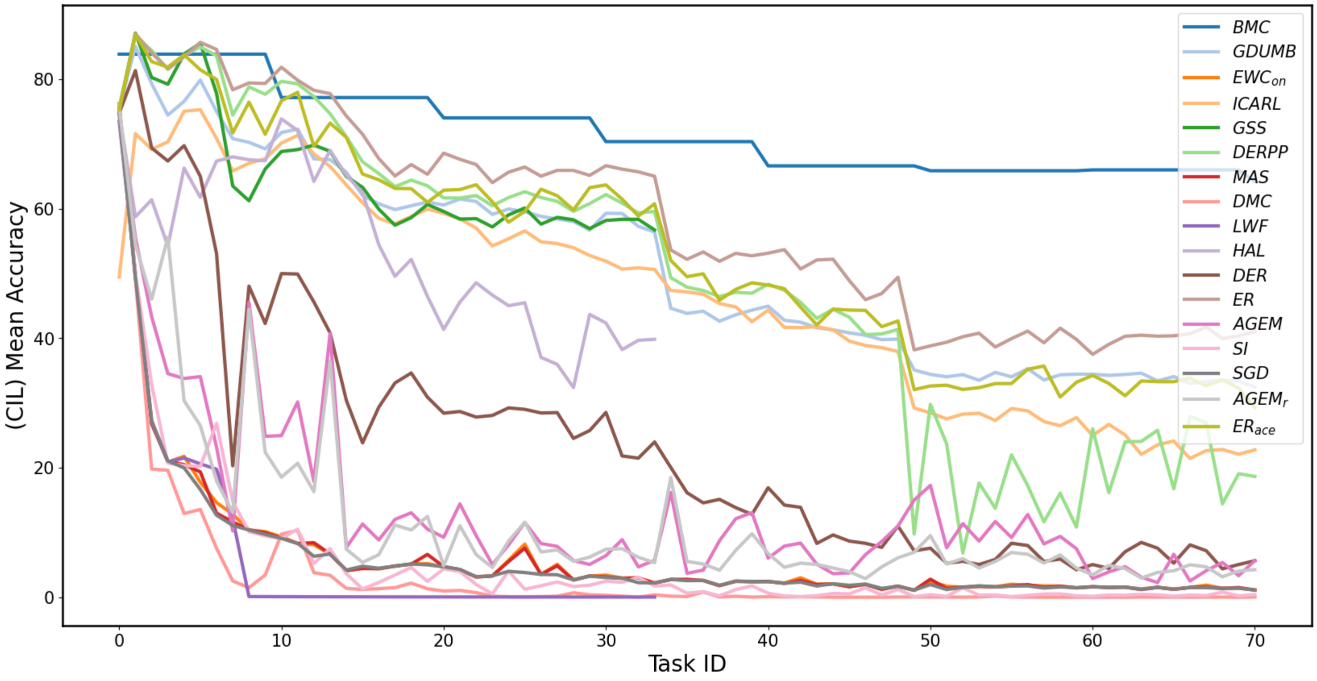


Figure 2. (CIL) Mean Accuracy, some methods (GSS, HAL, LwF) [85, 86, 88] fail to complete task 34 (iNaturalist [35]) due to the size of the dataset.

ments but for the same reasons is not suitable for a benchmark.

We run 629 experiments of 128 tasks and randomly sample each hyper-parameter that controls a different component for our method, which require 1 week of training time on a GPU cluster of x8 V100. We vary the stability coefficient ($\lambda$), consolidation coefficient ($\beta$), Number of Experts, Consolidation Loss $\mathcal{L}_{bmc}$, Stability Loss $\mathcal{L}_{bd}$ and both Memory and Buffer sampling Method. Each experimental configuration is randomly sampled, and as such, it is important to consider both the mean and the best-performing configuration when evaluating a setting. The reason is that there can be poor synergy between two randomly sampled settings or that a method is not fully evaluated. For example, consider that a really small value for $\lambda$ can be used and as such the full effect of the loss function used for a stability loss cannot be evaluated in that context. Additionally, each Stability Loss can have different sensitivity to $\lambda$ and as such maintaining the coefficient fixed or evaluating on different ranges can make the comparison non-equivalent. All settings have their values randomly sampled from the reported interval. Results presented in Fig. 5, Fig. 6, Fig. 7 and discussed in this section.

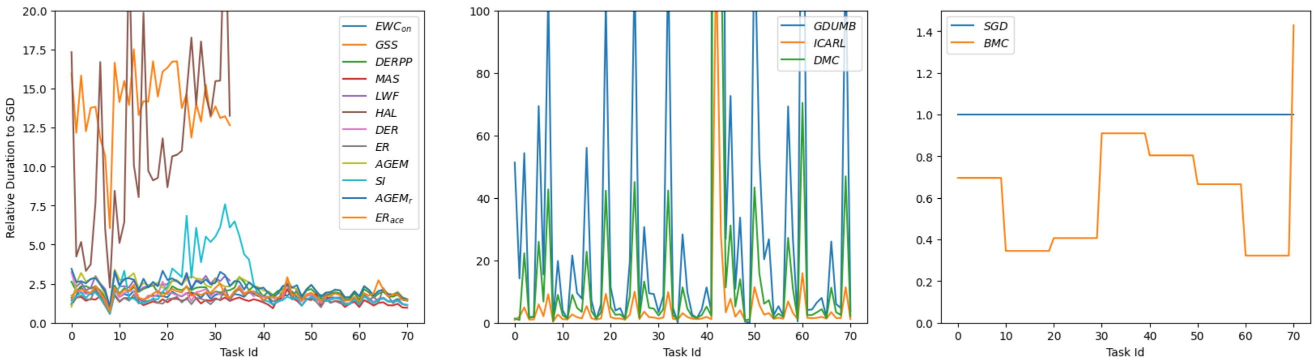


Figure 3. Relative time performance of each baseline compared to run-time using SGD (**Left**). A factor of 1 signifies equivalent performance to SGD. SGD performance is independent of the task dataset size, while some methods can have a component that depends on the size of the task dataset (**Middle**). Our method (**Right**) performs better than SGD under a distributed setting.

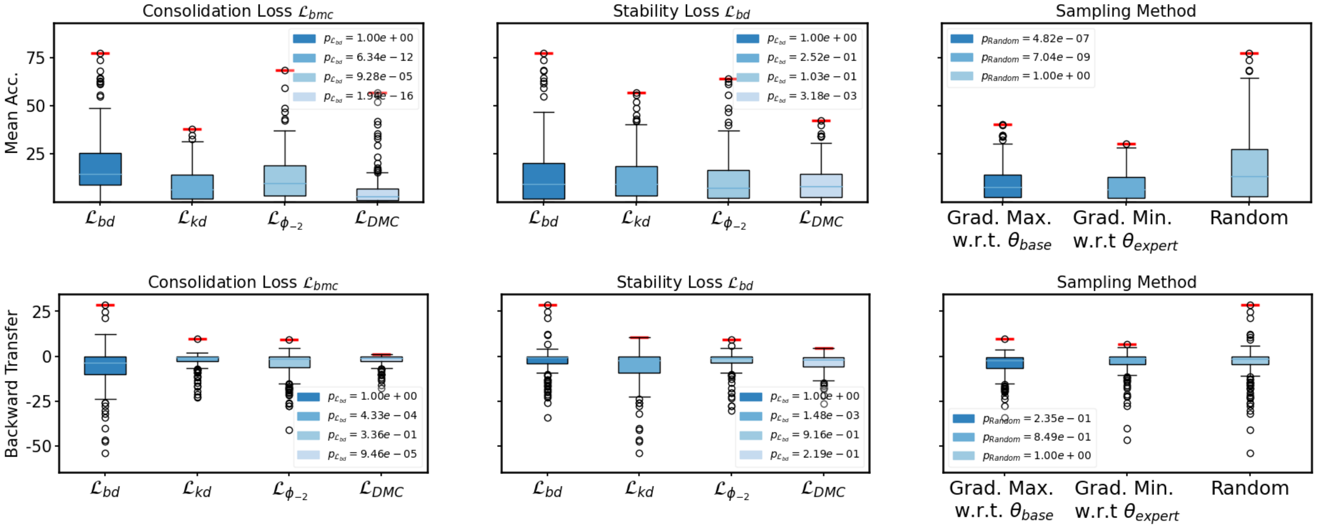


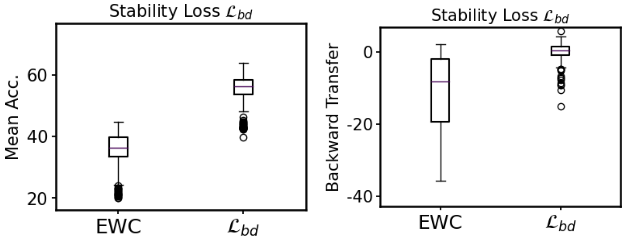
Figure 5. Aggregated results on an ablation study with 629 experiments for Permuted-MNIST. We randomly sample the type of loss for each phase of our method, the weight coefficient of each component as well as the sampling method. From Left to Right. Evaluation of loss in direct replacement to $\mathcal{L}_{bd}$ (Ours) for alternative losses $\mathcal{L}_{kd}$, $\mathcal{L}_{\phi_{-2}}$, $\mathcal{L}_{DMC}$. Stability Loss as a regularization component in replacement to $\mathcal{L}_{bd}$ and sampling method in replacement to Random. (**Top**) row reports the Mean Accuracy, where higher is better. (**Bottom**) row reports the Backward Transfer, where higher is better. $\mathcal{L}_{bd}$ has both the best-performing trials and a higher mean score for each metric. Naive random sampling for constructing the buffer performs the best. $p$ values are reported from one-way ANOVA between our purposed method and each corresponding method. Additional details in Appendix C.1, Appendix C.2 and Appendix C.3

### C.1. Batched Distillation Loss

We consider several alternatives in direct replacement to $\mathcal{L}_{bd}$. We use $\mathcal{L}_{bd}$ for two components of our method, on the Consolidation Loss for $\mathcal{L}_{bmc}$ and as a Stability Loss. We report the results in Fig. 5 (Left and Middle). We evaluate three alternative methods, such as $\mathcal{L}_{kd}$, $\mathcal{L}_{\phi_{-2}}$, $\mathcal{L}_{DMC}$. $\mathcal{L}_{kd}$ is Knowledge Distillation (KD) [92] applied on the logit space, similar to [81, 87]. $\mathcal{L}_{\phi_{-2}}$ is Knowledge Distillation applied on the pen-ultimate representation [93, 94]. $\mathcal{L}_{DMC}$ is Knowledge Distillation applied on a slice of logits using *double distillation loss* [73], similar to [95]. We evaluate the statistical significance on both evaluation metrics, Mean Accuracy and Backward Transfer. When considering the statistical significance on both metrics, $\mathcal{L}_{bd}$ outperforms other alternatives when used in $\mathcal{L}_{bmc}$ and as a Stability Loss. $\mathcal{L}_{\phi_{-2}}$, performs similarly to $\mathcal{L}_{bd}$ and is able to reach a higher Mean Accuracy in the study. However, we find that the results are not consistent and the mean value for $\mathcal{L}_{\phi_{-2}}$ on each metric is lower. The result is not statistically significant based on a $p$-value $> 0.05$. As such, the two methods can be evaluated further in future work.

## C.2. Regularization Loss

Elastic Weight Consolidation (EWC) [96] uses an alternative loss term to the current task loss that provides an optimization constraint on the parameters when training on a new task. The importance of each parameter to the current task is calculated based on an approximation of the Fisher Information Matrix. We use EWC as *stability-loss* in direct replacement for $\mathcal{L}_{bd}$. Figure 6 shows that EWC poses a strict constraint to the parameter and is unable to learn new tasks. $\mathcal{L}_{bd}$ outperforms EWC in this context. This could be explained by the large domain shift between each permutation and the limitation in the capacity of the backbone model for which it is not possible to isolate all parameters while learning new tasks. As such, we hypothesize that constructive interference methods such as KD are better candidates for both components of our method.



(a) Mean Accuracy

(b) Mean Forgetting

Figure 6. Comparison of using EWC in replacement to $\mathcal{L}_{bd}$ for Mean Accuracy (**Left**) and Backward Transfer (**Right**).

## C.3. Buffer Sampling

We examine two alternatives to Random sampling. We use gradient information as a heuristic when constructing the Buffer. In detail, we use the samples that produce the largest gradient with respect to the base model ($\theta_{base}$) with the intuition that they will be the most informative during the consolidation of $\theta_{base}$. We also use samples that produce the smallest gradient norms w.r.t. the expert model ($\theta_{expert}$) with the intuition that they are the most representative of the task the expert was trained on. Both methods perform poorly as compared to Random sampling. We hypothesize alternatives or improvements in the buffer sampling method can outperform Random in terms of task performance, but also consider that they can perform poorly in terms of run-time.

## C.4. Parameter Importance

We evaluate the most important component of our method using fANOVA parameter importance [97]. We find that the Number of Experts contributes the most to both the Mean Accuracy and Backward Transfer. Interestingly, both $\mathcal{L}_{bmc}$ and the sampling method contribute more to the Mean Accuracy than Backward Transfer. Our findings in Fig. 7 agree with our analysis that $\mathcal{L}_{bmc}$ provides a better approximation to the multi-task gradient as opposed to single-task consolidation and finally achieves higher Mean Accuracy within a batch of tasks. Likewise, a higher Number of Experts puts more constraints on the gradient updates than a small Number of Experts, making the gradient less 'sharp'. It reduces the bias toward every single task and benefits the Backward Transfer as it protects the parameters for previous tasks.

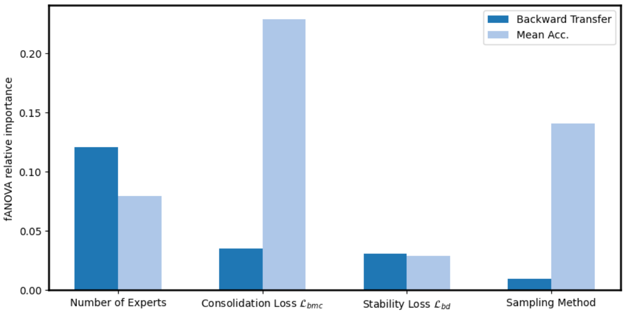


Figure 7. Importance of each component of our method in Backward Transfer (Dark Blue) and Mean Accuracy (Light Blue). We report the importance score using fANOVA [97]

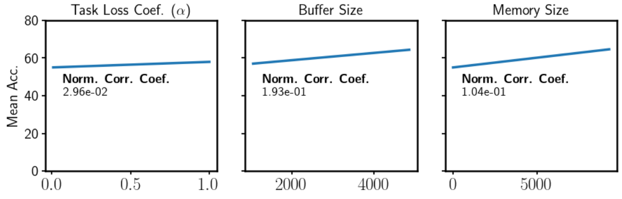


Figure 8. We vary the coefficient of task loss as well as the buffer and memory size to examine their effects on final average accuracy performance.

## C.5. Task Loss Coefficient & Buffer-Memory Sizes

We attach the ablation study results for task loss coefficient in the consolidation phase, buffer size and memory size in Fig. 8. Figures for buffer and memory size are as supplementary to the correlation we reported in the main text. In task loss coefficient, we find its low correlation to the final average accuracy.

## C.6. Backbone Model

Results obtained in Tab. 4 are subject to the backbone model used in extracting feature vectors. As we have to optimize 16 baselines; for 6,770,722 images and 2,866 classes, CLIP embeddings allow us to evaluate the merits of each baseline without extra computational cost. We include the

| Method | CLIP | ViT | ResNet50 | Avg. |
|---|---|---|---|---|
| ER | 41.4 | 32.8 | 27.6 | 33.9 |
| DER++ | 19.4 | 15.1 | 12.7 | 15.7 |
| **BMC** | **70.4** | **60.2** | **47.0** | **59.2** |

Table 5. Comparing BMC (ours), ER and DER++ on different backbone models CLIP, ViT [98] and ResNet50 on Stream Benchmark following the same experiment configurations. BMC outperforms other methods constantly on all backbones.

results of the ablation between the pre-trained CLIP, ViT and ResNet50 features on Stream Benchmark and show that our method can work across different backbones in Tab. 5. BMC outperforms the next best method (ER) by 25.3% on 3 backbones in average. We emphasize that the evaluation of the backbone model is orthogonal to both our method and the benchmark, as any backbone can be used in direct replacement.

| Task ID | Name | Num. Classes | Num. Train Images | Num. Val Images | Sub Task | Val. Acc. (SGD) |
|---|---|---|---|---|---|---|
| 0 | Aircraft [1] | 70 | 3334 | 3333 | Family | 74.74 |
| 1 | Apparel [2] | 6 | 8538 | 2847 | Color | 98.63 |
| 2 | Aptos2019 [3] | 5 | 2746 | 916 | - | 81.88 |
| 3 | Art [4] | 14 | 72009 | 24004 | - | 84.21 |
| 4 | Asl [5] | 29 | 65250 | 21750 | - | 99.89 |
| 5 | Boat [6] | 9 | 2193 | 731 | - | 99.86 |
| 6 | Cars [7] | 196 | 8144 | 8041 | - | 88.70 |
| 7 | Cataract [8] | 4 | 901 | 301 | - | 88.70 |
| 8 | CelebA [9] | 2 | 151949 | 50650 | Shadow | 93.42 |
| 9 | Colorectal [10] | 8 | 7500 | 2500 | - | 97.16 |
| 10 | Concrete [11] | 2 | 30000 | 10000 | - | 99.90 |
| 11 | Core50 [12] | 50 | 123649 | 41217 | Object | 99.69 |
| 12 | Cub [13] | 200 | 5994 | 5794 | - | 82.31 |
| 13 | Deepweedsx [14] | 9 | 15007 | 2501 | - | 93.24 |
| 14 | Dermnet [15] | 23 | 15557 | 4002 | - | 63.09 |
| 15 | Dtd [16] | 47 | 1880 | 1880 | Split 1 | 76.60 |
| 16 | Electronic [17] | 36 | 16152 | 5384 | - | 76.10 |
| 17 | Emnist [18] | 47 | 112800 | 18800 | Balanced | 86.61 |
| 18 | Eurosat [19] | 10 | 20250 | 6750 | - | 97.61 |
| 19 | Event [20] | 8 | 1180 | 394 | - | 100.00 |
| 20 | Face [21] | 3 | 10902 | 3634 | - | 98.98 |
| 21 | Fashion [22] | 5 | 33329 | 11112 | Gender | 94.47 |
| 22 | Fer2013 [23] | 7 | 28709 | 7178 | - | 72.99 |
| 23 | Fgvc6 [24] | 251 | 118475 | 11994 | - | 79.31 |
| 24 | Fish [25] | 9 | 6750 | 2250 | - | 100.00 |
| 25 | Flowers [26] | 17 | 1020 | 340 | Split 1 | 99.12 |
| 26 | Food101 [27] | 101 | 75750 | 25250 | - | 95.03 |
| 27 | Freiburg [28] | 25 | 3710 | 1237 | - | 97.33 |
| 28 | Galaxy10 [29] | 10 | 13302 | 4434 | - | 77.60 |
| 29 | Garbage [30] | 12 | 11636 | 3879 | - | 98.20 |

Table 6. Dataset used in Stream benchmark. Where applicable, Sub Task refers to the dataset split used in classifying each dataset. Additional details in Appendix A.1 and additional dataset in Tab. 7.

| Task ID | Name | Num. Classes | Num. Train Images | Num. Val Images | Sub Task | Val. Acc. (SGD) |
|---|---|---|---|---|---|---|
| 30 | Gtsrb [31] | 43 | 39209 | 12630 | - | 94.13 |
| 31 | Ham10000 [32] | 7 | 15022 | 5008 | - | 95.09 |
| 32 | Handwritten [33] | 33 | 1237 | 413 | Letters | 74.09 |
| 33 | Histaerial [34] | 7 | 26460 | 11340 | Small | 75.26 |
| 34 | iNaturalist [35] | 51 | 2686843 | 100000 | Species | 96.36 |
| 35 | Indoor [36] | 67 | 5360 | 1340 | - | 92.46 |
| 36 | Intel [37] | 6 | 14034 | 3000 | - | 95.90 |
| 37 | Ip02 [38] | 102 | 52603 | 22619 | - | 70.14 |
| 38 | Kermany2018 [39] | 4 | 83516 | 968 | - | 97.00 |
| 39 | Kvasircapsule [40] | 14 | 28342 | 9448 | - | 97.52 |
| 40 | Landuse [41] | 21 | 9450 | 1050 | - | 99.05 |
| 41 | Lego [42] | 46 | 32000 | 8000 | - | 91.20 |
| 42 | Malacca [43] | 3 | 121 | 41 | - | 100.00 |
| 43 | Manga [44] | 7 | 341 | 114 | - | 76.32 |
| 44 | Minerals [45] | 7 | 4111 | 1371 | - | 93.87 |
| 45 | Office [46] | 65 | 1820 | 607 | Art | 84.68 |
| 46 | Oriset [47] | 4 | 11110 | 3703 | Origami | 95.52 |
| 47 | Oxford [48] | 17 | 3797 | 1266 | - | 66.03 |
| 48 | Pcam [49] | 2 | 262144 | 32768 | - | 82.06 |
| 49 | Places365 [50] | 365 | 1803460 | 36500 | - | 54.79 |
| 50 | Planets [51] | 11 | 1228 | 410 | - | 100.00 |
| 51 | Plantdoc [52] | 28 | 2340 | 236 | - | 61.86 |
| 52 | Pneumonia [53] | 2 | 5232 | 624 | - | 81.09 |
| 53 | Pokemon [54] | 150 | 4994 | 1665 | - | 95.62 |
| 54 | Products [55] | 12 | 59551 | 60502 | - | 88.10 |
| 55 | Resisc45 [56] | 45 | 23625 | 7875 | - | 95.77 |
| 56 | Rice [57] | 5 | 56250 | 18750 | - | 99.84 |
| 57 | Rock [58] | 7 | 1515 | 506 | - | 82.21 |
| 58 | Rooms [59] | 5 | 3937 | 1313 | - | 93.37 |
| 59 | Rvl [60] | 16 | 320000 | 39999 | - | 88.03 |
| 60 | Santa [61] | 2 | 614 | 616 | - | 98.54 |
| 61 | Satellite [62] | 4 | 35679 | 11894 | - | 94.97 |
| 62 | Simpsons [63] | 42 | 31399 | 10467 | - | 99.38 |
| 63 | Sketch [64] | 250 | 15000 | 5000 | - | 78.82 |
| 64 | Sports [65] | 100 | 14072 | 500 | - | 99.00 |
| 65 | Svhn [66] | 10 | 73257 | 26032 | - | 82.23 |
| 66 | Textures [67] | 64 | 4335 | 4340 | - | 99.82 |
| 67 | Vegetable [68] | 15 | 18000 | 3000 | - | 99.93 |
| 68 | Watermarked [69] | 2 | 24987 | 6588 | - | 95.39 |
| 69 | Weather [70] | 4 | 841 | 281 | - | 98.22 |
| 70 | Zalando [71] | 6 | 24270 | 8090 | - | 78.44 |

Table 7. Dataset used in Stream benchmark. When applicable, Sub Task refers to the split of the dataset used in classifying each dataset. Additional details in Appendix A.1 and additional dataset in Tab. 6.

## References

[1] Subhransu Maji, Esa Rahtu, Juho Kannala, Matthew B. Blaschko, and Andrea Vedaldi. Fine-grained visual classification of aircraft. *CoRR*, abs/1306.5151, 2013. 1, 8

[2] Aleksandr Antonov. Apparel images dataset. https://www.kaggle.com/datasets/trolukovich/apparel-images-dataset. Accessed: 2022-10-30. 1, 8

[3] Asia Pacific Tele-Ophthalmology Society (APTOS). APTOS 2019 blindness detection. https://www.kaggle.com/competitions/aptos2019-blindness-detection/overview. Accessed: 2022-10-30. 1, 8

[4] 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. 1, 8

[5] 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. 1, 8

[6] 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. 1, 8

[7] 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. 1, 8

[8] jr2ngb (username). Cataract dataset. https://www.kaggle.com/datasets/jr2ngb/cataractdataset. Accessed: 2022-11-10. 1, 8

[9] 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. 1, 8

[10] 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. 1, 8

[11] Ç. F. Özgenel. Concrete crack images for classification. *Mendeley Data*, V2, 2019. 1, 8

[12] 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. 1, 8

[13] 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. 1, 8

[14] 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. 1, 8

[15] 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. 1, 8

[16] 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. 1, 3, 8

[17] 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. 1, 8

[18] Gregory Cohen, Saeed Afshar, Jonathan Tapson, and André van Schaik. EMNIST: an extension of MNIST to handwritten letters. *CoRR*, abs/1702.05373, 2017. 1, 8

[19] 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. 1, 8

[20] 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. 1, 8

[21] 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. 1, 8

[22] 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. 1, 8

[23] Manas Sambare. Fer-2013: Learn facial expressions from an image. https://www.kaggle.com/datasets/msambare/fer2013. Accessed: 2022-11-10. 1, 8

[24] 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. 1, 8

[25] 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. 1, 8

[26] 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. 1, 8

[27] Lukas Bossard, Matthieu Guillaumin, and Luc Van Gool. Food-101 – mining discriminative components with random forests. In *European Conference on Computer Vision*, 2014. 1, 8

[28] Philipp Jund, Nichola Abdo, Andreas Eitel, and Wolfram Burgard. The freiburg groceries dataset. *CoRR*, abs/1611.05799, 2016. 1, 8

[29] 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. 1, 8

[30] Mostafa Mohamed. Garbage classification (12 classes): Images dataset for classifying household garbage. https://www.kaggle.com/datasets/mostafaabla/garbage-classification. Accessed: 2022-11-10. 1, 8

[31] 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. 1, 9

[32] 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. 1, 9

[33] Olga Belitskaya. Classification of handwritten letters: Images of russian letters. https://www.kaggle.com/datasets/olgabelitskaya/classification-of-handwritten-letters. Accessed: 2022-11-10. 1, 9

[34] 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. 1, 9

[35] Visipedia. inaturalist 2021 competition: Fgvc8 workshop at cvpr. https://github.com/visipedia/inat_comp/tree/master/2021. Accessed: 2022-10-30. 1, 2, 4, 9

[36] Ariadna Quattoni and Antonio Torralba. Recognizing indoor scenes. In *2009 IEEE conference on computer vision and pattern recognition*, pages 413–420. IEEE, 2009. 1, 9

[37] Puneet Bansal. Intel image classification: Image scene classification of multiclass. https://www.kaggle.com/datasets/puneet6060/intel-image-classification. Accessed: 2022-11-10. 1, 9

[38] 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. 1, 9

[39] 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. 1, 9

[40] Manish KC (username). The kvasir-capsule dataset. https://www.kaggle.com/datasets/manishkc06/the-kvasircapsule-dataset. Accessed: 2022-11-10. 1, 9

[41] 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. 1, 9

[42] 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. 1, 9

[43] 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. 1, 9

[44] 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. 1, 9

[45] YoucefATTALLAH97 (username). Minerals identification & classification: Minet v2. https://www.kaggle.com/datasets/youcefattallah97/minerals-identification-classification. Accessed: 2022-11-10. 1, 9

[46] 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. 1, 9

[47] 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. 1, 9

[48] 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. 1, 9

[49] 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. 1, 9

[50] 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. 1, 9

[51] 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. 1, 9

[52] 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. 1, 9

[53] Paul Mooney. Chest x-ray images (pneumonia). https://www.kaggle.com/datasets/

paultimothymooney/chest-xray-pneumonia. Accessed: 2022-11-10. 1, 9

[54] Lance Zhang. 7000 hand-cropped and labeled Pokemon images for classification. https://www.kaggle.com/datasets/lantian773030/pokemonclassification. Accessed: 2022-11-10. 1, 9

[55] 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. 1, 9

[56] 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. 1, 9

[57] Murat Koklu, Ilkay Cinar, and Yavuz Selim Taspinar. Classification of rice varieties with deep learning methods. *Computers and electronics in agriculture*, 187:106285, 2021. 1, 9

[58] 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. 1, 9

[59] RobinReni (username). House rooms image dataset. https://www.kaggle.com/datasets/robinreni/house-rooms-image-dataset. Accessed: 2022-11-10. 1, 9

[60] 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. 1, 9

[61] 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. 1, 9

[62] 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. 1, 9

[63] Alexandre Attia. Simpson recognition. https://github.com/alexattia/SimpsonRecognition. Accessed: 2022-11-10. 1, 9

[64] Mathias Eitz, James Hays, and Marc Alexa. How do humans sketch objects? *ACM Trans. Graph. (Proc. SIGGRAPH)*, 31(4):44:1–44:10, 2012. 1, 9

[65] Gerry (username). 100 sports image classification. https://www.kaggle.com/datasets/gpiosenka/sports-classification. Accessed: 2022-11-10. 1, 9

[66] 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. 1, 9

[67] Yibin Huang. Textures classification dataset. https://github.com/abin24/Textures-Dataset. Accessed: 2022-11-10. 1, 9

[68] 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. 1, 9

[69] Felice Pollano. Watermark dataset builder. https://github.com/FelicePollano/WatermarkDataSetBuilder. Accessed: 2022-11-10. 1, 9

[70] A Gbeminiyi. Multi-class weather dataset for image classification. *Mendeley Data*, 2018. 1, 9

[71] 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. 1, 9

[72] 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

[73] 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. 1, 2, 3, 5

[74] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456. PMLR, 2015. 1

[75] Abien Fred Agarap. Deep learning using rectified linear units (relu). *arXiv preprint arXiv:1803.08375*, 2018. 1

[76] Philipp Moritz, Robert Nishihara, Stephanie Wang, Alexey Tumanov, Richard Liaw, Eric Liang, William Paul, Michael I. Jordan, and Ion Stoica. Ray: A distributed framework for emerging AI applications. *CoRR*, abs/1712.05889, 2017. 1

[77] Sebastian Ruder. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*, 2016. 1

[78] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 249–256. JMLR Workshop and Conference Proceedings, 2010. 1

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

[80] Arslan Chaudhry, Marcus Rohrbach, Mohamed Elhoseiny, Thalaiyasingam Ajanthan, Puneet K Dokania, Philip HS Torr, and Marc'Aurelio Ranzato. On tiny episodic memories in continual learning. *arXiv preprint arXiv:1902.10486*, 2019. 2

[81] 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, 5

[82] 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

[83] 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. 2

[84] 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. 2, 3

[85] 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, 3, 4

[86] 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. 2, 3, 4

[87] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, and Christoph H. Lampert. icarl: Incremental classifier and representation learning. *CoRR*, abs/1611.07725, 2016. 2, 3, 5

[88] Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE transactions on pattern analysis and machine intelligence*, 40(12):2935–2947, 2017. 2, 4

[89] 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. 2

[90] Friedemann Zenke, Ben Poole, and Surya Ganguli. Continual learning through synaptic intelligence. In *International Conference on Machine Learning*, pages 3987–3995. PMLR, 2017. 2, 3

[91] Zifeng Wang, Zizhao Zhang, Chen-Yu Lee, Han Zhang, Ruoxi Sun, Xiaoqi Ren, Guolong Su, Vincent Perot, Jennifer G. Dy, and Tomas Pfister. Learning to prompt for continual learning. *CoRR*, abs/2112.08654, 2021. 2

[92] Geoffrey Hinton, Oriol Vinyals, Jeff Dean, et al. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2(7), 2015. 5

[93] Jing Yang, Brais Martinez, Adrian Bulat, and Georgios Tzimiropoulos. Knowledge distillation via softmax regression representation learning. In *International Conference on Learning Representations*, 2021. 5

[94] 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. 5

[95] Antonio Carta, Andrea Cossu, Vincenzo Lomonaco, and Davide Bacciu. Ex-model: Continual learning from a stream of trained models. *CoRR*, abs/2112.06511, 2021. 5

[96] 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. 6

[97] Frank Hutter, Holger Hoos, and Kevin Leyton-Brown. An efficient approach for assessing hyperparameter importance. In *International conference on machine learning*, pages 754–762. PMLR, 2014. 6

[98] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. *CoRR*, abs/2010.11929, 2020. 7