ImpressLearn: Continual Learning via Combined Task Impressions

Anonymous authors
Paper under double-blind review

Abstract

This work proposes a new method to sequentially train a deep neural network on multiple tasks without suffering catastrophic forgetting, while endowing it with the capability to quickly adapt to unseen tasks. Starting from existing work on network masking (Wortsman et al., 2020), we show that simply learning a linear combination of a small number of task-specific masks (*impressions*) on a randomly initialized backbone network is sufficient to both retain accuracy on previously learned tasks, as well as achieve high accuracy on new tasks. In contrast to previous methods, we do not require to generate dedicated masks or contexts for each new task, instead leveraging transfer learning to keep per-task parameter overhead small. Our work illustrates the power of linearly combining individual impressions, each of which fares poorly in isolation, to achieve performance comparable to a dedicated mask. Moreover, even repeated impressions from the same task (homogeneous masks), when combined can approach the performance of heterogeneous combinations if sufficiently many impressions are used. Our approach scales more efficiently than existing methods, often requiring orders of magnitude fewer parameters and can function without modification even when task identity is missing. In addition, in the setting where task labels are not given at inference, our algorithm gives an often favorable alternative to the entropy based task-inference methods proposed in (Wortsman et al., 2020). We evaluate our method on a number of well known image classification data sets and architectures.

1 Introduction

Sequential learning without catastrophic forgetting has been an area of active research in machine learning for some time (Maes et al., 1996; Thrun & Pratt, 1998; Serra et al., 2018). A precondition for achieving Artificial General Intelligence is that models should be able to learn and remember a wide variety of tasks sequentially, without forgetting previously learned ones. In real-world scenarios, data from different tasks may not be available simultaneously, which makes it imperative to both allow continued learning of a potentially unbounded number of tasks (see also *The Sequential Learning Problem* (McCloskey & Cohen, 1989), *Constraints Imposed by Learning and Forgetting Functions* (Ratcliff, 1990) and *Lifelong Learning Algorithms* (Thrun & Pratt, 1998)). Recently, some successful approaches to combat this problem use task specific sub-models, which allow neural networks to context-switch between different learning tasks (Wortsman et al., 2020; Mallya et al., 2018; Mancini et al., 2018). The underlying context for each task can be represented as "*filters*" or "*masks*", altering the network connections for each task. Yet all of these approaches scale unfavorably with the number of unique tasks to be learned.

ImpressLearn We propose a novel method which exploits transfer learning and network masking to sequentially learn a theoretically unlimited number of tasks with much lower parameter overhead compared to prevailing benchmarks. Our method, termed *ImpressLearn*, uses elements from *Supermasks in Superposition (SupSup)* by Wortsman et al. (2020), which leverages the observation that even within randomly weighted neural networks there exist task-specific supermasks—subnetworks produced by overlaying a binary mask that selectively removes connections—which achieve good performance on the task. These supermasks can be learned from task-specific data and stored, one mask per task. At inference, the appropriate task-specific mask is applied when task identity is known. When task-labels of a previously seen task are not known, the correct mask can be inferred via entropy considerations.

Our ImpressLearn algorithm dramatically improves scaling of the parameters with the number of different tasks by reducing the number of necessary masks to a constant number, independent of the number of upcoming tasks. Our *basis-masks* are constructed from a small number of initial tasks (heterogeneous setting), or from just a single task (homogeneous setting). Leveraging transfer learning, this set of learned basis-masks, each of which can be interpreted as an impression of a previously seen basis-task, serves as a collection of latent features for new learning objectives, encoding common structural information. This might be reminiscent of associative learning where impressions of previous scenarios are combined to cope with new ones. Once a set of basis-masks is identified, we learn an appropriate linear combination of these impressions to quickly construct a real-valued mask that performs well on a new task. Hence, apart from a fixed number of basis-masks, only a small number of coefficients need to be learned and stored for each subsequent task. This greatly benefits scalability, a major drawback of previous methods (Wortsman et al., 2020; Mallya et al., 2018). In principle, this allows for an unlimited number of new tasks and, when the number of tasks grows, the parameter overhead is orders of magnitude smaller than required to store separate binary masks for every task.

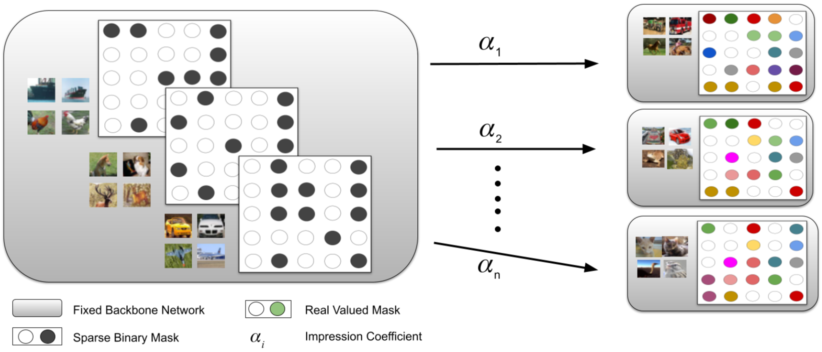

Figure 1: Overview of the ImpressLearn method (best viewed in colour). Given a fixed-weight randomly initialized backbone network, we show that a few binary mask "impressions" from training prior tasks (3 in this case) can form a linear combination with coefficients α to learn new tasks. Unlike the impressions, the resulting mask is real-valued, and not binary.

Homogeneous and random basis-masks Somewhat surprisingly, we can even generate all basis-masks from the same initial task using different random seeds for the learning algorithm (but the same random backbone network). We show that with a sufficiently large number of such homogeneous impressions, our algorithm learns linear combinations with close to benchmark accuracy on new tasks. We are reminded of an infant learning by taking different “snapshots” of the same object to infer properties of another. This homogeneous setting is particularly useful to address possible drifts in the data; akin to ensembling, it leverages the power of linear combinations for transfer learning. An additional important advantage is that the homogeneous setting has no limit on the number of basis-masks we can generate ab initio (since this requires a single task only).

To provide another baseline for our approach, we experimented with optimizing for a linear combination of entirely random masks of desired sparsity. We demonstrate on several benchmarks that if we chose a sufficiently large collection of such random basis-masks, our optimization still yields good performance. While combinations of random masks naturally lag behind the heterogeneous and the homogeneous settings, we show that there is a trade-off between the number of masks and their task-specificity (non-randomness). In settings where producing task-specific basis-masks is costly, optimizing for a linear combination of a large number of random masks can still yield satisfactory results.

Example: LeNet-300-100 on RotMNIST As a sneak-peak of our approach and its performance, Figure 2 shows the accuracy of ImpressLearn compared to SupSup on Rotated MNIST task set. First, as a sanity check, we apply basis-masks obtained from one task to tasks they were not optimized for. As expected, this yields essentially random accuracy (see X in Figure 2), confirming that the performance of ImpressLearn is beyond pure transfer learning and comes from linearly combining the initial impressions. Next, Figure 2 illustrates that ImpressLearn even with a small number of heterogeneous basis-masks is on par or even superior to SupSup on unseen tasks. Note that ImpressLearn with a basis set of 10 masks requires only $3 \times 10 = 30$ parameters; in contrast, SupSup needs to generate an entire binary mask, which requires $\geq 25K$ tensor indices to specify (assuming 10% sparsity). Figure 2 also illustrates the performance of ImpressLearn over homogeneous basis-masks; in this setting, we need a larger number of basis-masks to achieve accuracy comparable to the heterogeneous scenario. Ultimately, however, we still match the performance of SupSup with a vastly smaller number of additional per-task parameters. Lastly, the right plot of Figure 2 shows that our algorithm can successfully operate when task identity is not provided at inference (cf. GN regime (Wortsman et al., 2020)). Here, our optimization procedure finds the "correct" basis-mask or a linear combination of basis-masks yielding similar or better performance, providing an alternative to the entropy-based GN-inference developed by Wortsman et al. (2020). In Section 4, we provide more empirical evidence of the efficacy of ImpressLearn on a variety of benchmarks, outlining the radical savings in parameters that need to be stored per task compared to SupSup.

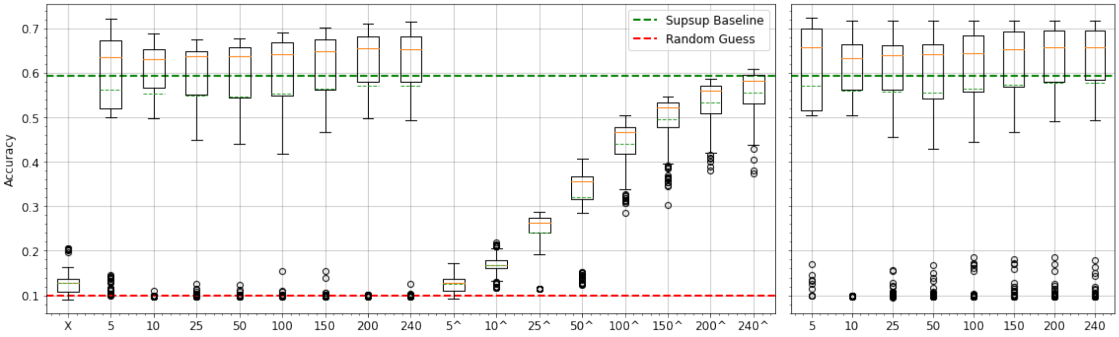

Figure 2: Left box plot: Average validation performance across 10 unseen RotatedMNIST tasks by the number of basis impressions and impression type (X—incorrect mask, n—the number of basis heterogenuous impressions, $n^\wedge$—the number of basis homogeneous impressions). Right box plot: Average performance across seen tasks in the GN regime (no task identity available at inference). All results are averaged over 3 different seeds and over masks of sparsity $\in \{0.95, 0.92, 0.9, 0.85, 0.8, 0.75, 0.7\}$

The GN regime Both SupSup and ImpressLearn are required to have access to task identities in order to apply the right mask to the backbone network. To relax this requirement, Wortsman et al. (2020) extended their algorithm to a more challenging regime (coined *GN*—Given/Not given) where task identifiers are present during training but unavailable at inference. To achieve this, (Wortsman et al., 2020) use a one-shot minimization of the entropy of the model's outputs to single out the correct mask. In Section 4, we show that our ImpressLearn algorithm is also able to achieve this task. In particular, we demonstrate that, when applied to basis-tasks in the GN regime, the optimization routine of ImpressLearn either identifies the corresponding basis-mask or finds a better performing combination of basis-masks.

In Section 2, we briefly review related work and general approaches to countering catastrophic forgetting, highlighting research that motivated our approach. In Section 3, we lay out the ImpressLearn algorithm, including our modified objective function. In Section 4, we demonstrate the effectiveness of ImpressLearn on a variety of datasets and architectures to show close-to-benchmark performance with a drastically reduced parameter count on unseen tasks, especially where the number of possible tasks is large. We discuss our results including the trade-offs between homogeneous and heterogeneous masks. Finally, in Section 5, we talk about limitations of our work and avenues for future research.

2 Related Work

In practice, intelligence systems should be able to learn a variety of tasks incrementally and without experiencing catastrophic forgetting—degrading performance on previously acquired skills (McCloskey & Cohen, 1989), while ideally transferring current knowledge to facilitate future training. Oftentimes tasks are diverse and not available concurrently, making joint training impractical. Conversely, ordinary finetuning (continued training of a pre-trained network) inevitably leads to catastrophic forgetting. Continual learning encompasses a broad spectrum of algorithms and architectures that address these issues and propose systems capable of learning from an incremental stream of tasks while minimizing catastrophic forgetting. Most naturally, these techniques are categorized into three groups described below (Delange et al., 2021; Wortsman et al., 2020).

Regularization-based methods This class of algorithms trains on new tasks by finetuning weights but attempts to retain performance on previously learned tasks through regularization. A number of studies assess the importance of individual parameters for previous tasks and penalize their displacement accordingly during optimization. Pioneering this approach, Serra et al. (2018) estimate parameter importance using a Laplace-approximated posterior distribution after training on earlier tasks. Zenke et al. (2017) impose a quadratic penalty proportional to the accumulated sensitivity of previous loss functions to perturbations in the corresponding parameter; Aljundi et al. (2018) use the same strategy but accumulate sensitivity of the network output to parameter perturbations instead. In contrast, Li & Hoiem (2018) regularize by means of distilling current knowledge on the incoming task's data and using it during finetuning. Regularization-based methods require no additional memory overhead per task and hence are advantageous in capacity constrained settings. However, the plasticity of a network decreases with more tasks, imposing a natural limit on new tasks, which creates a trade-off between learning new tasks and catastrophic forgetting.

Replay methods Techniques in this category preserve performance on prior tasks by replaying, rehearsing or otherwise utilizing representative samples from the corresponding data distributions. Most commonly, replay models store examples of seen data in a separate memory buffer (Rebuffi et al., 2017; Rolnick et al., 2019; Riemer et al., 2019); others maintain generators that approximate the original data distribution and provide pseudo-examples (Atkinson et al., 2021; Shin et al., 2017). While the majority of algorithms in this group replay stored examples during optimization to mimic joint training, Lopez-Paz & Ranzato (2017) use them to constrain optimization space and ensure positive knowledge transfer. Replay methods require additional memory to store data samples or allocate generators, however, these costs are usually kept fixed. For this reason, like regularization-based models, replay models exhibit poor stability-plasticity trade-off with more tasks and often come with increased memory requirements when compared to regularization-based methods.

Parameter isolation methods These methods allocate new parameters for incoming tasks and feature little to no interference between previously learned tasks. Rusu et al. (2016) allocate a new copy of the network and enable forward transfer learning with lateral connections going into new modules. Ren et al. (2017) combine individual learners in a decision tree and eliminate outdated models with tree pruning. A large body of recent algorithms piggyback on a single backbone network shared by all tasks. As such, Wen et al. (2020) (BatchEnsemble) operate on a fixed pretrained network and, for each incoming task, optimize for a rank one parameter mask applied to the backbone at inference. Mallya & Lazebnik (2018) (PackNet) use pruning to assign subsets of free parameters of a backbone network to individual tasks by issuing one binary parameter mask per task. These assigned parameters are forever frozen at their trained values, limiting the capacity of the network for future tasks. In a subsequent study, Mallya et al. (2018) (Piggyback) lift this limitation by directly optimizing per-task binary masks and applying them to a fixed pretrained network.

Supermasks in Superposition (SupSup) ImpressLearn is most closely related to yet another related method called SupSup (Wortsman et al., 2020). This algorithm trains individual per-task binary masks and applies them to a randomly-initialized network, leveraging the existence of supermasks (Zhou et al., 2019). The mask optimization algorithm, edge-popup (Ramanujan et al., 2019), uses a heaviside function to binarize mask values on the forward pass and employs a straight-through estimator when computing gradients. In addition, Wortsman et al. (2020) propose different training and inference modes depending on availability of task identifiers; e.g., GG refers to the scenario when task identifies are known during both training and inference, while in GN they are available only during training. In the latter case, Wortsman et al. (2020) introduce a one-shot algorithm to infer task identity by minimizing entropy of the output, starting from a uniform linear combination of masks and optimizing the coefficients. While at first glance this algorithm resembles ours, there are essential differences as we use optimization of a refined linear combination to learn *new* tasks. While SupSup and other related methods suffer no catastrophic forgetting regardless of the number of tasks, they are required to store the corresponding parameter masks for each task, which is costly. (Wortsman et al., 2020) address this by storing masks as attractors of a Hopfield network, but it is unclear how feasible this approach is.

3 Approach

Preliminaries and notation We largely adopt the notation from Wortsman et al. (2020). For the standard l-way classification task from a set of tasks T, inputs x are mapped to a distribution p over output neurons $\{1, ..., l\}$. Let f be a network architecture defined over the backbone weight matrix W, which is taken to be random but fixed. Similar to (Wortsman et al., 2020) we use the Edge-Popup training algorithm of Ramanujan et al. (2019) (based on an earlier work by Zhou et al. (2019)) to train a binary mask M^t for a task $t \in T$. We further stratify each mask by layer: let d be the number of weight-layers of the network computing f. For $i \in [1, \dots, d]$ denote by M^t_i the part of the binary matrix corresponding to layer i of the network such that $M^t = \oplus_i M^t_i$. Similarly, let W_i denote the submatrix of W corresponding to the ith layer. The *sparsity* $\in (0, 1]$ of a mask M^t is given by the fraction of zeroes in the mask and is usually fixed in advance to s. For each task $t \in T$ SupSup finds a mask M^t to minimize $E_{(x,y)\sim\mathcal{D}}\mathcal{L}(y, f(x, M^t \odot W))$, where $\mathcal{L}$ is the loss function and $\mathcal{D}$ is the given data-generating distribution.

The ImpressLearn algorithm We define a set of basis-tasks $T_b \subset T$ and refer to its complement $T_n = T \setminus T_b$ as collection of new tasks. We randomly initialize and freeze the backbone network with weights W^1. Then, the algorithm proceeds as follows:

Step 1: For each $t \in T_b$ we use the edge-popup algorithm to create one or several basis-masks M^t, leading to a collection $\mathcal{M} = \{M^1, M^2, \dots\}$ of basis-masks. We use up to 250 basis-masks for various benchmark architectures and data sets.

Step 2: For each new task $s \in T_n$ we define a matrix $\alpha^s \in \mathbb{R}^{|\mathcal{M}|\times d}$ of learnable coefficients, one per basis-mask per layer. To find the layerwise linear combination of basis-masks, we use SGD for the following optimization problem:

$$\hat{\alpha}^s = \arg\min_{\alpha^s} \; \mathcal{L}\left(y, f\left(x, \textstyle\sum_{M^t\in\mathcal{M}} \oplus_{i=1}^{d} \alpha^s_{t,i}(M^t_i \odot W_i)\right)\right). \tag{1}$$

Initialization of α A priori all basis-masks have equal chance to contribute to learning of the new task. Moreover, testing any single basis-mask optimized for task t on any other task $t' \neq t$ gives almost random performance, implying that there is no direct knowledge transfer (see Section 4). Hence, a uniform prior on coefficients α is a reasonable assumption. We treat each layer independently, setting $\alpha_{i,t} = 1/|\mathcal{M}|$ at the start of the gradient-based optimization.

Regularization While overfitting is less of a concern, given the small number of parameters in our optimization equation (1), it is desirable that ImpressLearn discovers a sparse solution (i.e., uses as few basis-masks as possible). This is especially relevant when performing inference on basis-tasks, where ImpressLearn is expected to identify the "correct" mask among all basis-masks. Hence, in certain cases we apply regularization on parameters α for each layer to obtain the following loss function:

$$J = \mathcal{L} + \lambda \textstyle\sum_{i=1}^{d} \left(\sum_{t=1}^{|\mathcal{M}|} |\alpha^s_{t,i}| - 1\right)^2 \tag{2}$$

---

[1] We use Kaiming normal distribution (He et al., 2020) and set bias variables to zero.

Heterogeneous, homogeneous & random masks In our *heterogenous* approach, we create one basis-mask for each task in the collection of basis-tasks T_b. In this setting, we leverage knowledge transfer from all basis-tasks to a new task. However, when tasks are scarce, limiting basis-mask generation to one per task affects the viability of our method and does not allow for scaling benefits to become apparent. For example, for Split-CIFAR-100 with 20 tasks, we can only generate at most 20 heterogeneous basis-masks. Hence, we also evaluate our approach with a set of *homogeneous* basis-masks, i.e. all coming from the same basis-task. A priori it is unclear whether masks produced on the same backbone network for the same basis-task are sufficiently different to generate a diverse enough basis set. While previous work suggests that wider network architectures can support multiple suitable subnetworks for a given task Ramanujan et al. (2019); Frankle & Carbin (2019), we find this effect is prominent even using relatively conservative architectures such as LeNet-300-100 with PermutedMNIST (Lecun et al. (1998)). Our experiments show that basis-masks are very sensitive to initial conditions of the popup scores and to data ordering: the overlap of homogeneous masks produced with different random seeds is close to random. Hence, at sufficiently low sparsities, homogeneous masks are practically independent. ImpressLearn can be extended to a mix of homogeneous and heterogeneous impressions, though in this paper we only study the trade-off for purely homogeneous and purely heterogeneous settings.

To quantify the importance of task specific data in mask generation, we experimented with a variation of ImpressLearn where basis-masks are drawn randomly and not optimized (see Section 4 and B). This allows us to study trade-offs between random and task-specific masks and provide an alternative when optimizing basis-masks with edge-popup is too demanding. While in general this approach requires more basis-masks to reach commensurate performance, it may be more suitable in scenarios where optimization is costly.

Optimization and parallelization We use the edge-popup algorithm with random initializations for the edge popup scores and data ordering to generate basis-mask collection $\mathcal{M}$. One fetching attribute of ImpressLearn is the ability to parallelize the training process. Our approach allows for all masks to share the same backbone network, both in the heterogeneous and the homogeneous settings. Thus, one can train multiple basis-masks at the same time on different GPUs, only constrained by the number of cores or GPU accelerators available.

4 Experimental Results

We evaluated ImpressLearn over the following classification datasets: MNIST (Lecun et al., 1998): Permuted and Rotated; Split-CIFAR-100 (Krizhevsky, 2012) and Split-ImageNet (Deng et al., 2009). A detailed description of our experimental choices and infrastructure can be found in A.

Main results Overall, our experiments demonstrate the strength of ImpressLearn in various settings (Figures 3, 4). ImpressLearn is capable of successfully learning new tasks with a fraction of the parameters required by other approaches. In line with expectations, we see a strong positive relationship between accuracy and the size of the impression set $|\mathcal{M}|$, which saturates at different data-specific sizes. While the number of heterogeneous masks required for competitive accuracy varies by dataset, ImpressLearn is particularly resource-efficient when the number of possible tasks is large so that it becomes prohibitively costly to store a separate mask for each task (e.g. PermutedMNIST with 784! tasks). Compared to the heterogeneous scenario, we need more homogeneous basis-masks to achieve similar performance (and even more random masks, see B). For SplitImageNet, we only show results for a heterogeneous basis-mask set with $|\mathcal{M}| \leq 35$ due to limitations on compute (Figure 5). We anticipate that with a larger basis-mask set, ImpressLearn will achieve the performance of SupSup. Additionally, note that ImpressLearn's α-optimization procedure yields performance superior to SupSup's entropy minimization under the GN regime on SplitImageNet.

Ablation: incorrect masks We confirm that all performance of ImpressLearn comes not from chance application of a suitable mask that does well on other tasks but rather from appropriately combining multiple masks through a learned linear function. In particular, we test impressions derived from one basis-task on other tasks and find that the incorrect basis-mask fails to have any predictive power on other tasks when used

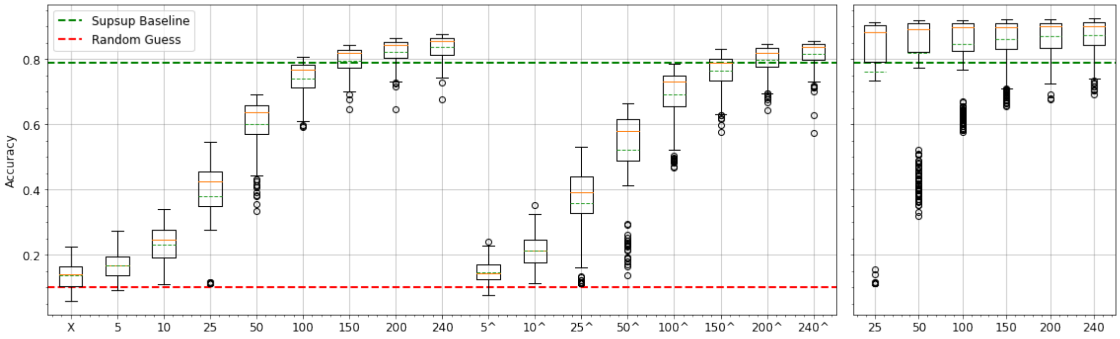

Figure 3: Left box plot: Average performance on 10 new `PermutedMNIST` tasks by the number of impressions and impression type (X—incorrect mask; $n^\wedge$—the number of homogeneous masks, n—the number of random masks). Right box plot: performance of our α-optimization in the GN regime restricted to basis-tasks. All results are averaged across 3 different seeds and masks of sparsity $\in \{0.95, 0.92, 0.9, 0.85, 0.8, 0.75, 0.7\}$.

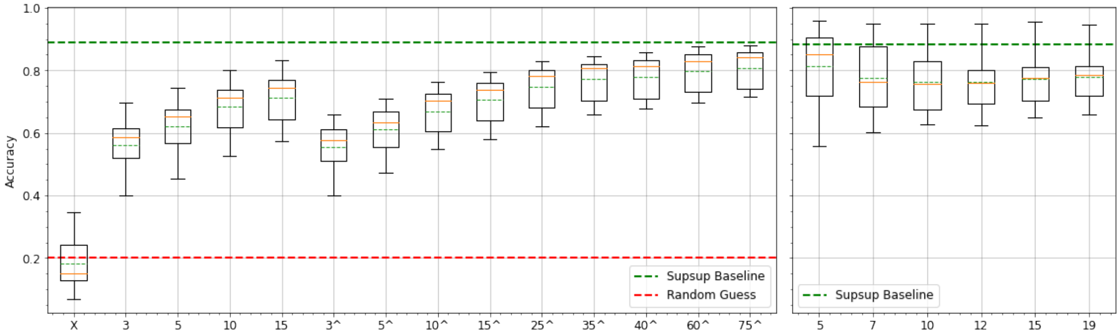

Figure 4: Left box plot: Average performance on 5 new Split-CIFAR-100 tasks by number of impressions and impression type (X—incorrect mask; $n^\wedge$—the number of homogeneous masks, n—the number of random masks). Right box plot: performance of our α-optimization in the GN regime restricted to basis-tasks. All results are averaged across 3 different seeds and masks of sparsity $\in \{0.95, 0.92, 0.9, 0.85, 0.8, 0.75, 0.7\}$.

in isolation, giving a roughly random accuracy of $10 \pm 3\%$ for Permuted/Rotated-MNIST/Split-ImageNet (Figures 2, 3, and 5) and $20 \pm 2\%$ on Split-CIFAR-100 (Figure 4), respectively.

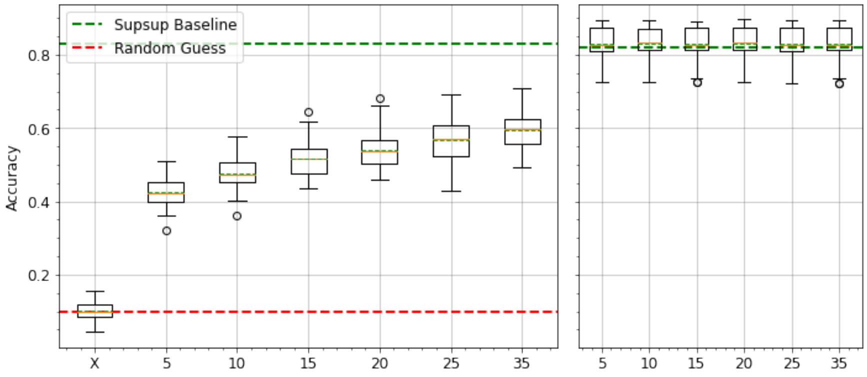

Figure 5: Left: Average performance on 5 new Split-ImageNet tasks by number of heterogeneous impressions; X indicates incorrect mask. Right: Performance of our α-optimization in the GN regime restricted to basis-tasks.

Ablation: hybrid ImpressLearn In order to understand the value of linear combination of basis-masks, we implemented a hybrid method that follows ImpressLearn in all hidden layers (i.e., optimizes a linear combination of basis-masks for each new task) and SupSup in the output layer (i.e., allocates a learned binary parameter-mask for each new task). This hybrid approach is only marginally better than ImpressLearn and only when the number of basis-masks is small. As the number of basis-masks increases, the two methods yield equivalent performance (Section C and Figure 10). This demonstrates that ImpressLearn is not just a simplified version of SupSup but rather a novel algorithm with its unique properties and core principles. Additionally, these ablation experiments show that ImpressLearn is unlike trivial finetuning of the classification layer; instead, it takes advantage of all layers to reach its performance.

GN regime To evaluate the strength of our optimization routine, we studied the regime where task identifiers are not provided at inference (GN). In the first set of experiments, we employ ImpressLearn's α-optimization and the one-shot entropy minimization from Wortsman et al. (2020) over a linear combination of all basis-masks with respect to one of the basis-tasks from T_b without explicitly providing its identity. For most architecture-dataset pairs, our strategy either determines the correct basis-mask or discovers an even better-performing linear combination of basis-masks (the right plots of Figures 2, 3, 4, and 5,). Thus, as shown in Figure 6, ImpressLearn applied to unknown basis-tasks outperforms SupSup in both GN and GG regimes, i.e., even when task labels are available to SupSup and not ImpressLearn. Similar results are observed when applying the two algorithms in the GN regime on a mixture of basis-masks and real-valued masks precomputed from the coefficients discovered by ImpressLearn for non-basis tasks. As before, our optimization routine compares favourably to the one-shot entropy minimization.

Model efficiency & parameter savings We present parameter savings of our ImpressLearn algorithm compared to SupSup across various architectures and datasets (Table 1). Our approach has the fixed cost of storing the basis-masks as well as the task-specific coefficients (α) for each new task. Amortizing mask storage across tasks and accounting for storage of the coefficients α, we get particularly impressive savings of an order of magnitude or more for datasets with a large number of tasks (e.g., PermutedMNIST and ImageNet). We have also evaluated potential savings our approach could yield for several common architectures by giving an educated guess on the number of required basis-masks. In the case of ImageNet, the number of potential tasks is so large that storage of basis-masks per new task is very small even if we double the number of basis-masks considered, and memory required to store the coefficients remains small. Overall, we believe that ImpressLearn affords considerable savings in memory at the expense of either no or vanishing loss in accuracy, which can be further mitigated by expanding the set of basis-masks.

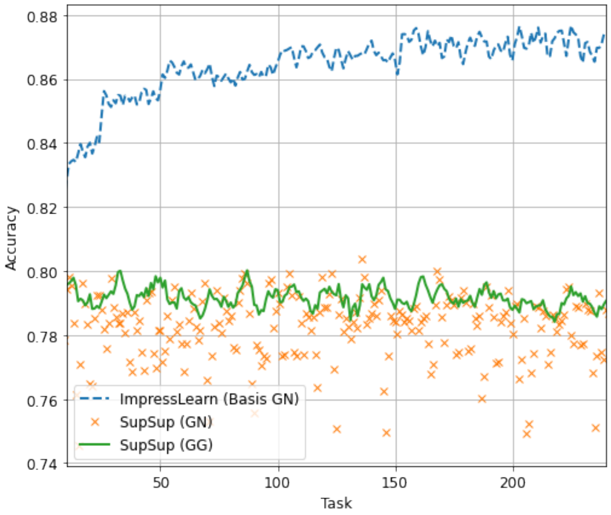

Figure 6: Validation accuracy of ImpressLearn-GN, SupSup-GG, and SupSup-GN (+one-shot entropy minimization) on PermutedMNIST. In the GN regime, results are averaged across 10 different data splits and orderings per seed and sparsity. All results are averaged across 3 different seeds and masks of sparsity $\in \{0.95, 0.92, 0.9, 0.85, 0.8, 0.75, 0.7\}$.

5 Discussion

In this work, we extend an existing continual learning algorithm (SupSup) and design our own (ImpressLearn) that, leveraging principles from transfer learning to generalize to new tasks, allows for scalable and parameter-efficient continual learning. Using a simple linear combination of masks, or impressions, we see that even this basic setup is capable of learning new tasks effectively. We show that this effect is consistent across task types and network architectures, and that it achieves competitive performance while using significantly fewer parameters. This work highlights the advantages of reusing existing meta-features learned on previous tasks for future learning problems and opens up a space of possibilities of applying transfer learning to protect against catastrophic forgetting.

Table 1: Parameter trade-off for different models and datasets. The maximum number of possible tasks for each dataset is given in brackets. For ImageNet, we have assumed 10-way classification tasks. For RotatedMNIST, we have assumed a 1 degree granularity of rotations. Paramweter counts do not include biases or batchnorm parameters. Mask size is the space on disk required to store each additional parameter mask as 16-bit integers. The number of basis-masks $|\mathcal{M}|$ was approximately chosen to give a close-to-benchmark performance on new tasks. The number of per-task parameters (floating point) required by ImpressLearn is denoted by $|\alpha^t|$. Φ is the storage per task amortizing the cost of storing $|\mathcal{M}|$ basis-masks over all possible tasks. Efficiency factor (eff.) is the compression or savings ratio with respect to SupSup.

Dataset (Max Tasks) Model	# params	mask (kB)	$\|\mathcal{M}\|$	$\|\alpha^t\|$	Φ (kB)	eff.
PermutedMNIST (784!)						
LeNet 300-100	266K	65	100	300	1	**55**
RotatedMNIST (359)						
LeNet 300-100	266K	65	5	15	1	**67**
Split-CIFAR-100 (20)						
ResNet-18	6.2M	1,513	10	210	757	**2**
Wide ResNet-18	11.7M	2,856	10	210	1,429	**2**
Wide ResNet-34	21.8M	5,322	10	370	2,662	**2**
ImageNET (2100)						
VGG-16	137.9M	33,667	75	1,200	1,207	**28**
ResNet-50	25.6M	6,250	75	3,975	238	**26**
ResNet-101	44.5M	10,864	75	7,800	418	**26**
ResNet-152	60.2M	14,697	75	11,625	570	**26**

One application of our approach could be protection against drifts in the data. In the homogeneous basis-masks setting, one could learn a few basis-masks as well as their linear combination, and then update the latter accordingly when anticipating that the underlying data distribution drifts steadily. To our knowledge, no other approach allows for such easy continual adjustment since most of them fix masks or weights for fixed tasks.

Limitations Our experimental results show that ImpressLearn works well on several benchmarks, and particularly shines when the number of new tasks is large. In scenarios where the number of different tasks is small (e.g., Split-CIFAR-100 with 20 tasks only), our approach will only give limited parameter savings, if any. For Split-ImageNet, we were only able to evaluate ImpressLearn for a relatively small number of masks and could not match benchmark performance. However, specifically in this case, our parameter savings are impressive and highlight the power of transfer learning in this setting.

References

Rahaf Aljundi, Francesca Babiloni, Mohamed Elhoseiny, Marcus Rohrbach, and Tinne Tuytelaars. Memory aware synapses: Learning what (not) to forget. In Vittorio Ferrari, Martial Hebert, Cristian Sminchisescu, and Yair Weiss (eds.), *Computer Vision – ECCV 2018*, pp. 144–161, Cham, 2018. Springer International Publishing. ISBN 978-3-030-01219-9.

Craig Atkinson, Brendan McCane, Lech Szymanski, and Anthony V. Robins. Pseudo-rehearsal: Achieving deep reinforcement learning without catastrophic forgetting. *Neurocomputing*, 428:291–307, 2021.

Matthias Delange, Rahaf Aljundi, Marc Masana, Sarah Parisot, Xu Jia, Ales Leonardis, Greg Slabaugh, and Tinne Tuytelaars. A continual learning survey: Defying forgetting in classification tasks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1–1, 2021. doi: 10.1109/TPAMI.2021.3057446.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 248–255, 2009.

Jonathan Frankle and Michael Carbin. The lottery ticket hypothesis: Finding sparse, trainable neural networks. In *International Conference on Learning Representations*, 2019.

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 9726–9735, 2020.

Alex Krizhevsky. Learning multiple layers of features from tiny images. *University of Toronto*, 05 2012.

Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(12):2935–2947, 2018. doi: 10.1109/TPAMI.2017.2773081.

David Lopez-Paz and Marc' Aurelio Ranzato. Gradient episodic memory for continual learning. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

Pattie Maes, Maja J. Mataric, Jean-Arcady Meyer, Jordan Pollack, and Stewart W. Wilson. *Incremental Self-Improvement for Life-Time Multi-Agent Reinforcement Learning*, pp. 516–525. The MIT Press, 1996.

Arun Mallya and S. Lazebnik. Packnet: Adding multiple tasks to a single network by iterative pruning. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7765–7773, 2018.

Arun Mallya, Dillon Davis, and Svetlana Lazebnik. Piggyback: Adapting a single network to multiple tasks by learning to mask weights. In Vittorio Ferrari, Martial Hebert, Cristian Sminchisescu, and Yair Weiss (eds.), *Computer Vision – ECCV 2018*, pp. 72–88, Cham, 2018. Springer International Publishing. ISBN 978-3-030-01225-0.

Massimiliano Mancini, Elisa Ricci, Barbara Caputo, and Samuel Rota Bulo. Adding new tasks to a single network with weight transformations using binary masks. In *Proceedings of the European Conference on Computer Vision (ECCV) Workshops*, September 2018.

Michael McCloskey and Neal J. Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. In Gordon H. Bower (ed.), *Psychology of Learning and Motivation*, volume 24, pp. 109–165. Academic Press, 1989.

Vivek Ramanujan, Mitchell Wortsman, Aniruddha Kembhavi, Ali Farhadi, and Mohammad Rastegari. What's hidden in a randomly weighted neural network? In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 11890–99, November 2019.

Roger Ratcliff. Connectionist models of recognition memory: Constraints imposed by learning and forgetting functions. *Psychological Review*, 97(2):285–308, 1990.

Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H. Lampert. icarl: Incremental classifier and representation learning. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5533–5542, 2017.

Boya Ren, Hongzhi Wang, Jianzhong Li, and Hong Gao. Life-long learning based on dynamic combination model. *Applied Soft Computing*, 56:398–404, 2017. ISSN 1568-4946.

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. In *In International Conference on Learning Representations (ICLR)*, 2019.

David Rolnick, Arun Ahuja, Jonathan Schwarz, Timothy Lillicrap, and Gregory Wayne. Experience replay for continual learning. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

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.

Joan Serra, Didac Suris, Marius Miron, and Alexandros Karatzoglou. Overcoming catastrophic forgetting with hard attention to the task. In Jennifer Dy and Andreas Krause (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 4548–4557. PMLR, 10–15 Jul 2018.

Hanul Shin, Jung Kwon Lee, Jaehong Kim, and Jiwon Kim. Continual learning with deep generative replay. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

Sebastian Thrun and Lorien Pratt. *Lifelong Learning Algorithms*. Springer US, Boston, MA, 1998.

Yeming Wen, Dustin Tran, and Jimmy Ba. Batchensemble: an alternative approach to efficient ensemble and lifelong learning. In *International Conference on Learning Representations*, 2020.

Mitchell Wortsman, Vivek Ramanujan, Rosanne Liu, Aniruddha Kembhavi, Mohammad Rastegari, Jason Yosinski, and Ali Farhadi. Supermasks in superposition. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS'20, Red Hook, NY, USA, 2020. Curran Associates Inc. ISBN 9781713829546.

Friedemann Zenke, Ben Poole, and Surya Ganguli. Continual learning through synaptic intelligence. In Doina Precup and Yee Whye Teh (eds.), *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 3987–3995. PMLR, 06–11 Aug 2017.

Hattie Zhou, Janice Lan, Rosanne Liu, and Jason Yosinski. Deconstructing lottery tickets: Zeros, signs, and the supermask. In *Advances in Neural Information Processing Systems*, 2019.

A Experimental details

Our experiments encompassed a range of sparsities: $s \in \{0.95, 0.92, 0.9, 0.85, 0.8, 0.75, 0.7\}$. For consistency, we maintained the same numerical ordering of tasks from each dataset across all experiments. Each seed varied the random initialization of edge-popup scores and the training data ordering. For a heterogeneous basis set, we used one seed to generate all basis-masks $\mathcal{M}$. In the homogeneous scenario, we seeded every mask to ensure mask diversity. The backbone network was kept fixed for each unique combination of $\mathcal{M}$ and α-optimization for new tasks. All runs were performed on three different backbone networks and train/test splits to ensure that results were sufficiently general. The boxplots show averages over all settings and sparsity levels.

For the training of all models except LeNet-300-100, we used GPU enabled hardware to expedite optimization. Our experiments were performed on an internal cluster enabled with NVIDIA V100 Tesla and RTX 8000 GPUs.

Table 2: Overview of hyperparameters. For homogeneous basis-masks, $|\mathcal{M}_{hom}|$ is the largest number of masks tried. For ADAM, momentum was set to 0.9 and weight decay to 0.1.

Model	dataset	$\|\mathcal{M}_{hom}\|$	LR	λ	optimizer	batch
LeNet-300-100	RotatedMNIST	250	0.002	0	RMSprop	128
LeNet-300-100	PermutedMNIST	250	0.002	0	RMSprop	128
ResNet-18	Split-CIFAR-100	75	0.02	0.005	ADAM	64
ResNet-50	Split-ImageNet	75	0.0025	0.005	ADAM	96

B Random Basis-masks

Here, we compare performance of a set of random basis-masks to homogeneous masks to illustrate the trade-offs when basis-mask optimization with edge-popup is prohibitively expensive (Figs. 7, 8 and 9). Generally, as the number of random masks increases, this modification of ImpressLearn fares surprisingly well and could become an alternative for resource-constrained scenarios with only a slight degradation in accuracy.

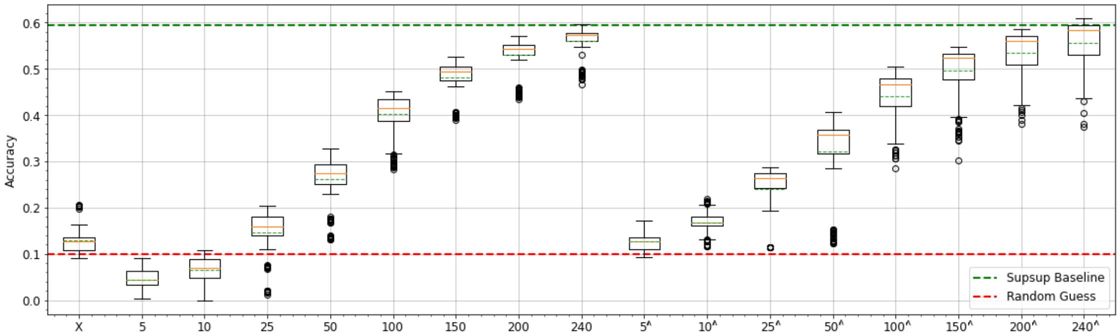

Figure 7: **Random vs homogeneous masks:** Average validation performance on 10 unseen RotatedMNIST tasks by number of impressions and impression type (X—incorrect mask; $n^\wedge$—the number of homogeneous masks, n—the number of random masks). All results averaged over 3 different seeds and masks of sparsity $\in \{0.95, 0.92, 0.9, 0.85, 0.8, 0.75, 0.7\}$.

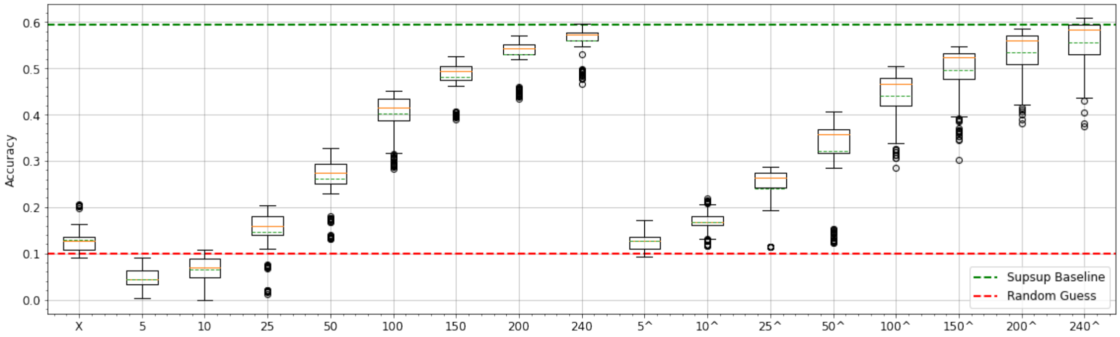

Figure 8: **Random vs homogeneous masks:** Average validation performance on 10 unseen PermutedMNIST tasks by number of impressions and impression type (X—incorrect mask; $n^\wedge$—the number of homogeneous masks, n—the number of random masks). All results averaged over 3 different seeds and masks of sparsity $\in \{0.95, 0.92, 0.9, 0.85, 0.8, 0.75, 0.7\}$.

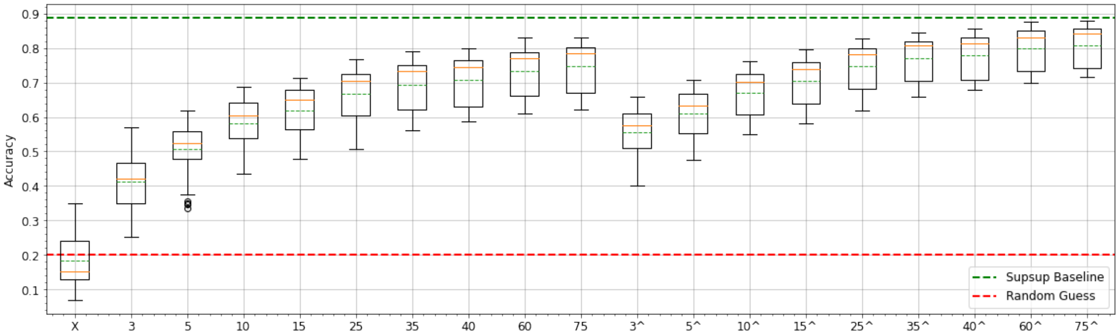

Figure 9: **Random vs homogeneous masks:** Average validation performance on 5 unseen Split-CIFAR-100 tasks by number of impressions and impression type (X—incorrect mask; $n^\wedge$—the number of homogeneous masks, n—the number of random masks). All results averaged over 3 different seeds and masks of sparsity $\in \{0.95, 0.92, 0.9, 0.85, 0.8, 0.75, 0.7\}$.

C Hybrid ImpressLearn

As another ablation experiment, we employ ImpressLearn for all hidden layers of the model but learn a separate binary parameter-mask for the output layer with edge-popup for each incoming task, just as SupSup does. As the number of basis impressions grows, the difference between this hybrid approach and our ImpressLearn vanishes (Figure 10). Hence, allowing more parameters in the last layer (for an additional edge-popup mask) does not improve the performance and the power of ImpressLearn does not reside solely in finetuning parameters for new tasks. Lastly, these results evidence that ImpressLearn is qualitatively different from SupSup and offers its own benefits and trade-offs.

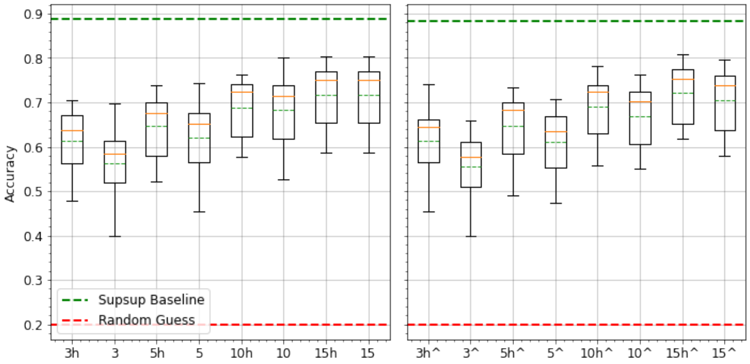

Figure 10: Left: hybrid (marked h) vs ImpressLearn (heterogeneous impressions). Right: hybrid (marked $h^\wedge$) vs ImpressLearn (homogeneous impressions). Average performance across 5 new Split-CIFAR-100 tasks by number of impressions, model, and impression type. All results averaged over 3 different seeds and masks of sparsity $\in \{0.95, 0.92, 0.9, 0.85, 0.8, 0.75, 0.7\}$.