# KNOWLEDGE ACCUMULATION IN CONTINUALLY LEARNED REPRESENTATIONS AND THE ISSUE OF FEATURE FORGETTING

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

While it is established that neural networks suffer from catastrophic forgetting "at the output level", it is debated whether this is also the case at the level of representations. Some studies ascribe a certain level of innate robustness to representations, that they only forget minimally and no critical information, while others claim that representations are also severely affected by forgetting. To settle this debate, we first discuss how this apparent disagreement might stem from the coexistence of two phenomena that affect the quality of continually learned representations: knowledge accumulation and feature forgetting. We then show that, even though it is true that feature forgetting can be small in absolute terms, newly learned information is forgotten just as catastrophically at the level of representations as it is at the output level. Next we show that this feature forgetting is problematic as it substantially slows down knowledge accumulation. We further show that representations that are continually learned through both supervised and self-supervised learning suffer from feature forgetting. Finally, we study how feature forgetting and knowledge accumulation are affected by different types of continual learning methods.

## 1 INTRODUCTION

Machine learning models typically learn from static datasets and once they are trained and deployed, they are usually not updated anymore. Sometimes models make mistakes. Sometimes they do not work in a domain that was not trained. Sometimes they do not recognize certain classes or corner cases. Whatever the cause, sometimes it is necessary to update a model. The default choice in industry is to retrain a model from the beginning with new and old data to overcome malfunctions (Komolafe, 2023). Retraining a full model is costly and time-consuming, especially in deep learning. The goal of continual learning is to enable models to train continually, to learn from new data when they become available. This has proven to be a hard challenge (De Lange et al., 2022; van de Ven et al., 2022), as deep learning models that are continually trained exhibit catastrophic forgetting (McCloskey & Cohen, 1989; Ratcliff, 1990). Without precautionary measures, new data are learned at the expense of forgetting earlier acquired knowledge.

The data to train machine learning models rarely come in a format that is adapted to the problem we intend to solve. Taking the example of visual data, it is near impossible to infer higher-level properties directly from an image's raw pixel values. Hence, a first step is usually to transform them into a *representation* that makes solving the problem at hand an easier job. Often deep neural networks are used for this (Bengio et al., 2013). These networks learn semantically meaningful representations indirectly while optimizing their parameters to learn an input-output mapping. Sometimes the representation itself is the goal, yet often it is a final layer, or head, that uses the learned representation to assign an output (*e.g.* a class label) to an input. Even though they are commonly trained in unison, it can be useful to think of the representation and the head as two separate entities, working together.

In continual learning, there are at least two good reasons to care about representations. First, a strong representation makes it easier to learn new information. When a model already has a good representation of new data, it will require less changes to fully adapt to new data. This makes the re-use of existing features more likely, in turn lowering the risk of overwriting them, which

increases the risk of forgetting (Cha et al., 2021). Second, progressively accumulating knowledge from individual tasks into one representation may be a goal on its own. True continual learning should be able to use new information to its benefit and build a stronger representation over time, which can finally be used to solve a variety of tasks (Bengio et al., 2013).

It is with these motivations that recent work has been studying how representations are learned in continual learning, and how they forget. Among the researched topics are the effect of the depth of a layer on forgetting and learning (Ramasesh et al., 2021; Kim & Han, 2023) and the apparent robustness of representations to forgetting (Davari et al., 2022; Zhang et al., 2022). These works offer interesting insights, but they do not agree and open questions remain. Davari et al. (2022) write: "*[...] in many commonly studied cases of catastrophic forgetting, the representations under naive finetuning approaches, undergo minimal forgetting, without losing critical task information.*" and Zhang et al. (2022) similarly write: "*there seems to be no catastrophic forgetting in terms of representations*". Yet in similar experimental setups Kim & Han (2023) identify "*severe catastrophic forgetting*".

Another open question concerns whether feature forgetting, if it happens, hinders the learning of good representations. When studying the representation of continual learners using a downstream task (*i.e*. one that was not trained), Zhang et al. (2022) conclude that "*learning representations and catastrophic forgetting are largely separate issues*" and "*common techniques for mitigating catastrophic forgetting [...] have little effect on improving [representations]*". Similar conclusions are drawn by Cha et al. (2022). This suggests that only task-specific features might be forgotten. If this were true, feature forgetting would only be a problem if you care about the performance on the trained tasks, but not if you care about learning a good general representation. Yet, in the same papers, it is shown that learning many tasks together results in a better general representation than learning those same tasks one after the other, which seems to contradict that only task-specific features are forgotten.

Given these unresolved issues in the literature about forgetting and learning in continual representations, we aim to answer two questions:

**Question 1**: *Do continually trained representations forget catastrophically?*

With extensive experiments, we show that also at the level of representations, when training on new tasks, that what was learned during a past task is abruptly and greatly forgotten, or as it is called in literature: catastrophically. This leads us to the follow-up question:

**Question 2**: *Does it matter that these representations are forgotten?*

To test the impact of feature forgetting on the quality of the continually learned representation for downstream tasks, we compare the representation of a continually trained model against a representation that is ensembled from copies of the model after it is trained on each task. This ensemble baseline has a substantially better general representation than the continually trained model, showing that preventing feature forgetting is not only important for the performance on tasks that a model was trained on, but also for optimal knowledge accumulation.

Most experiments in this paper study the learning and forgetting of representations in supervised learning, but we show that our answers to the above two questions also hold for representations learned with self-supervised learning. We conclude the paper by evaluating examples of important families of continual learning methods and report how they influence the learning and forgetting of representations.

In summary our contributions include[1]:

- We show that continually learned respresentations do forget catastrophically (Section 3).
- We show that such forgetting in the representation negatively affects knowledge accumulation (Section 4).
- We compare feature forgetting and knowledge accumulation in different types of continual learning methods (Section 5).
- We show that self-supervised and contrastive learners suffer from feature forgetting as well (Section 5).

[1]Code will be made public upon acceptance.

## 2 PROBLEM STATEMENT AND EVALUATION

We follow the common definition of a continual learning setting by assuming a stream $\mathcal{T} = \{\mathcal{T}_1, \mathcal{T}_2, ..., \mathcal{T}_T\}$ of $T$ disjoint tasks $\mathcal{T}_i$. Each task consists of training data $X_i$ and targets $Y_i$, as well as respective test data $\hat{X}_i$, $\hat{Y}_i$. During training on each task the model has free access to the training data of that task, but not to the data of other tasks. Exception are replay memories, which can store small subsets of data from past tasks. On this stream of tasks we continually train a model $f_\theta$, with the goal to learn a model that works well for all tasks. Because, in this work we are particularly interested in how models continually learn and adapt a representation from sequence $\mathcal{T}$, we split the model into a shared backbone that produces the representation with parameters $\theta_B$, and a head with task-specific parameters $\theta_H = \{\theta_{h_1}, ..., \theta_{h_T}\}$ that utilizes the representation to solve the tasks.

Our main focus is on classification tasks. To measure continual learning performance in the standard way, we define $\text{ACC}_{i,j}$ as the test accuracy (the percentage of correctly classified test samples) on task $\mathcal{T}_j$ obtained by the model after training on task $\mathcal{T}_i$. We refer to this as *output* accuracy. Additionally, and central to this work, we explicitly evaluate the quality of the continually learned representations. Inspired by representation learning literature (Bengio et al., 2013; Chen et al., 2020; Zeiler & Fergus, 2014), we define the metric *linear probe accuracy*, denoted $\text{LP}_{i,j}$. After finishing training on task $\mathcal{T}_i$, a new set of parameters $\theta_{h_j}$ for the head's parameters of task $\mathcal{T}_j$ are first trained with all training data in $\mathcal{T}_j$ while the backbone parameters $\theta_B$ are frozen. $\text{LP}_{i,j}$ is the test accuracy of the resulting model on task $\mathcal{T}_j$. The metric $\text{LP}_{i,j}$ thus measures the true suitability of the model's representation with respect to task $\mathcal{T}_j$ after training up to task $\mathcal{T}_i$. Lastly, when evaluating on a downstream task, *i.e.* one that was not part of training, this is indicated by $\text{LP}_{i,d}$. There are other ways to evaluate representations, *e.g.* using $k$-Nearest Neighbours, which we briefly review in the Supplemental. This did not lead to different conclusions, hence reporting in the main paper uses linear probes, as is common in the related literature.

In the main paper, the reported results are on Split MiniImageNet, a 20 task (5 classes each) split of MiniImageNet (Vinyals et al., 2016). The first 19 tasks are used as the training sequence, while the remaining task is never seen during training and used exclusively as a downstream task, to evaluate the quality of the representation. To reduce the influence of the inherent difficulty of a particular task, we use five different task sequences and report mean and standard errors on all results. The sequences are randomly generated but consistent across experiments. In the Supplemental material, we replicate all our results using a 10 task sequence of CIFAR-100 (Krizhevsky et al., 2009). For more details, see Supplemental.

## 3 REPRESENTATIONS FORGET CATASTROPHICALLY

To answer whether representations forget catastrophically, we need to comprehend what *"catastrophically"* refers to. For this, we turn to the two works that are often credited for discovering the phenomenon of catastrophic forgetting. McCloskey & Cohen (1989) note: "*[t]raining on a new set of items may drastically disrupt performance on previously learned items*", and Ratcliff (1990) describes this as: "*well-learned information is forgotten rapidly as new information is learned*". To be considered 'catastrophic', forgetting should thus be both 'drastic' and 'rapid'. We further note that, implicitly, both of the above descriptions consider the information that was learned during a task as what can be forgotten, respectively: "*previously learned items*" and "*well-learned information*". This is perhaps most clear in the definition of Robins (1993), inspired by the two earlier works: "*[i]f after its original training is finished a network is exposed to the learning of new information, then the originally learned information will typically be greatly disrupted or lost*". Summarized, catastrophic forgetting refers to the drastic and rapid forgetting of previously learned knowledge.

In recent papers, following Lopez-Paz & Ranzato (2017), forgetting is often calculated as the difference in performance immediately after training task $i$ and after training on a final task $n$. With $r_{i,j}$ the performance of task $j$ after training on task $i$, this becomes: $r_{i,i} - r_{n,i}$. They do not quantify how much forgetting would be considered catastrophic, and neither will we, but we can compare forgetting in the representation to that at the output layer, which is often identified as catastrophic. Note that $r$ can refer to any performance measure, so both output accuracy (ACC) and linear probing accuracy (LP) are a valid option. Forgetting, defined as such, only depends on the difference in performance relative to immediately after a task was trained, regardless of how much information

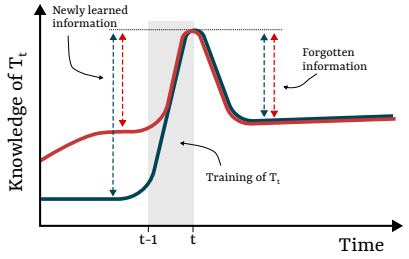



Figure 1: An illustration of why it matters to account for the learned information when calculating forgetting. Without one could conclude that both examples forget an equal amount. While the red example actually forgets everything it had learned, and the blue one only about 50%.

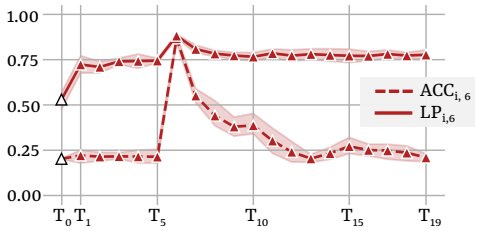



Figure 2: Linear probe and output accuracy of $T_6$ during the entire Mini-ImageNet sequence. $T_0$ indicates at model initialization, so before any training took place. (Mean $\pm$ standard error)

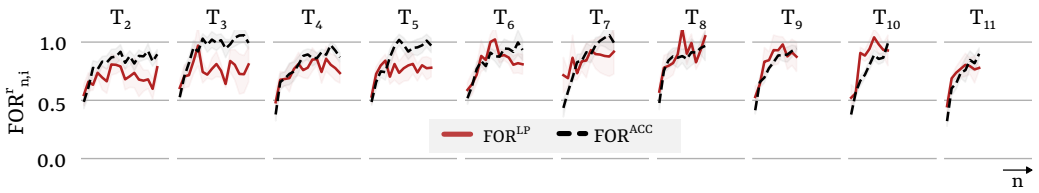



Figure 3: Forgetting at the output level ($FOR^{ACC}$) and at the level of representations ($FOR^{LP}$), as calculated by Equation 1, on the Mini-ImageNet sequence. When expressed as proportion of the knowledge gained during training on the task, forgetting in the representation is just as catastrophic as forgetting at the output.(Mean $\pm$ standard error)

was learned during training the task. Figure 1 shows why this matters. When not accounting for the initial performance, both examples in this figure forget an equal amount. While when we consider forgetting as the proportion of gained information that was lost, the red example forgets much more. To account for this, we propose to define *relative forgetting* of task $i$ after $n$ new tasks as:

$$\text{FOR}^r_{n,i} = \frac{r_{i,i} - r_{i+n,i}}{r_{i,i} - r_{i-1,i}} \tag{1}$$

Or, in words: relative forgetting is the proportion of knowledge that was gained during training on a task that is then lost after further training on other tasks. When comparing the output accuracy of two continual learning algorithms, our proposed way of measuring forgetting does not often lead to different conclusions, as before training on a task the accuracy is typically low or at chance level. However, for measuring forgetting in representations, our proposal is crucial. To evaluate a representation, some supervised information is always used, hence the initial accuracy will not be zero, but depends on the quality of the representation. While random performance will not change, the quality of the representation can, complicating the analysis further. It is comparable to the difference between the answers to the following two questions before seeing any data: "*Which test samples belong to the unknown category x?*" and "*Given that x looks like this, which other test samples are of category x?*". While the first answer will be random, the second one depends on how good the description, *e.g.* the representation, of *x* is.

In Figure 2, the output accuracy and linear probe accuracy of $T_6$ are shown throughout training. They both peak just after training the task, after which they decrease to a level close to the performance just before the task was trained. Importantly, the two aforementioned differences between output and probing accuracy are apparent in this plot. First, the baseline performance (indicated by $\Delta$) is much higher for the LP measure than for the ACC measure. Secondly, while the output accuracy on $T_6$ does not change by training on the first five tasks, the representation quality does increase. The exact proportions of gained knowledge that are forgotten are hard to compare in this figure, so in

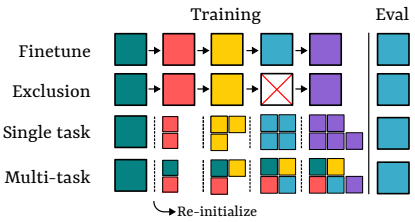



Figure 4: Illustration of the tasks that are trained at each stage for the baselines in Figure 5. Finetune and exclusion continue from the result of the previous task, the others are re-initialized. The white box means no task was trained.

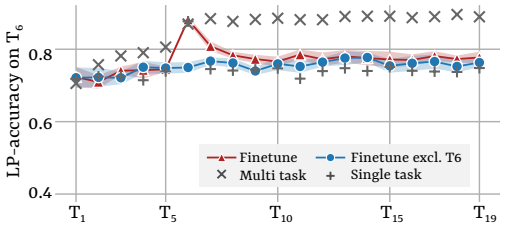



Figure 5: LP-accuracy ($LP_{i,6}$) of a naive finetuning baseline, the exclusion, single and multi-task baselines on teh Mini-ImageNet sequence. (Mean $\pm$ standard error).

| Excluded task ($T_e$) | $T_1$ | $T_6$ | $T_{11}$ | $T_{16}$ |
|---|---|---|---|---|
| Finetune | $74.6 \pm 3.2$ | $77.7 \pm 1.6$ | $76.7 \pm 2.5$ | $78.4 \pm 0.9$ |
| Exclusion | $75.2 \pm 2.7$ | $76.3 \pm 1.3$ | $74.8 \pm 2.1$ | $75.3 \pm 1.6$ |
| Single Task | $72.7 \pm 3.7$ | $74.8 \pm 0.8$ | $73.0 \pm 2.9$ | $71.4 \pm 2.5$ |
| Multi Task | $87.3 \pm 2.7$ | $89.0 \pm 1.2$ | $88.6 \pm 1.7$ | $89.4 \pm 1.3$ |

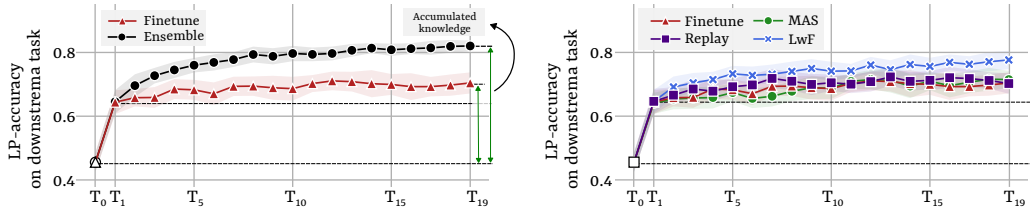
Table 1: Probing accuracy ($LP_{19,e}$ with $e$ the excluded task) at the end of training for tasks mentioned in the columns. Comparing finetuning with exclusion, single task and multi-task training. *e.g.* the second column reports the final values in Figure 5. (Mean $\pm$ standard error)

Figure 3, we show the relative forgetting for both the representation and the output, calculated using Equation 1. For every task $i$, $\text{FOR}_{1,i}$ is as high for the representation as for the output accuracy. For $\text{FOR}_{n,i}$, $n > 1$, it depends on when the task was trained. For early tasks, forgetting of the probe stabilizes, while the output continues to get worse. For the later tasks (see Supplemental for task 12 and more), the representation forgets at least as much as at the output.

Preventing forgetting is one goal of continual learning, forward and positive backward transfer are another: information from one task ideally improves the performance on earlier and later tasks. For representations forgetting and backward transfer can co-occur. Learning from a new task can make a model forget, but at the same time new information can also transfer to an old task. In some sense, this makes the result in Figure 3 only a lower bound to forgetting. It is possible that there is more forgetting, but transfer from other tasks improves the performance at the same time, negating some of the forgetting. Apart from transfer to other tasks, longer optimization with strong augmentations might also cause a model to learn a better representation. To estimate the contribution from transfer from other tasks, we train a model on the same sequence but *without* the evaluated task. This model cannot forget, but it has the transfer from other tasks. Similarly, we train a second model only on the latest task, but increase the number of iterations to match those of the sequential model to evaluate the influence of longer optimization. See Figure 4 for an illustration of their training processes. Figure 5 shows the results and Table 1 contains detailed results with more excluded tasks. Both models that were trained on a sequence of tasks outperform the single task baseline, showing that there is some benefit from training on multiple tasks. This is *knowledge accumulation*: small transfers from other tasks make the final representation better. The exclusion and finetune baseline finally reach nearly the same representation quality. The former cannot forget, so their similarity indicates that in the end, it did not matter much whether a task was trained or not, and a lot of information was forgotten.

## 4 Feature forgetting reduces knowledge accumulation

To answer the second question, whether or not representation forgetting is a problem, we want a baseline that learns in the same way as a continually finetuned model, but that has no forgetting. To achieve this, inspired by Vogelstein et al. (2020) and Yan et al. (2021), we design the *ensemble* baseline. This baseline stores a model copy after every task and concatenates the representation of all these models during evaluation, on top of which the probes are trained. The compute required for




(a) The ensemble baseline accumulates more knowledge than a finetune baseline.

(b) The tested methods of Section 5. LwF has less forgetting, which is confirmed by the better result.

Figure 6: LP-accuracies ($LP_{i,d}$) on a downstream task of Mini-ImageNet. (Mean $\pm$ standard error)

doing inference with this baseline increases linearly with every new task, but it allows us to study the no-forgetting scenario. See Supplemental for more details. Figure 6a shows the probing accuracy on a downstream task. While finetuning accumulates some knowledge, the ensemble baseline clearly accumulates more. We stress again that the finetune and the ensemble baseline learn in the exact same way, the only difference is that the ensemble baseline does not forget. These results thus show that knowledge accumulation is substantially reduced by feature forgetting. A potentially confounding factor in these experiments is that the dimension of the concatenated representation is higher than of the finetuned representation. To control for this, in the Supplemental we use PCA to reduce the dimensionality of the concatenated representation, and show that this does not change our conclusion.

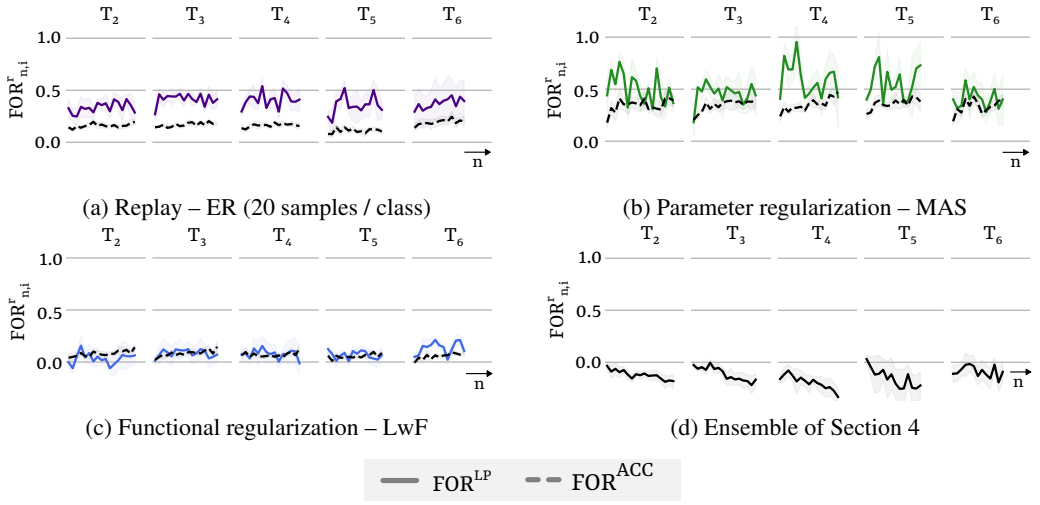
Learning many tasks together resutls in a better general represenation than learning those same tasks sequentially (*e.g.* Zhang et al., 2022; Cha et al., 2022. While it seems that this observation also implies that feature forgetting reduces knowledge accumulation, that conclusion is not actually justified from the observation. It is possible that the representation learned by joint multitask training is better than the representation of finetuning, not because of the absence of forgetting, but because training is done on all tasks at the same time. That is, joint multitask training and finetuning differ not only in terms of forgetting, but also in terms of how they learn.

Continually accumulating knowledge can be a goal on its own, and is often difficult to achieve. Recent works (Janson et al., 2022; Kim & Han, 2023) have shown that recent successful methods for continual learning rely on a pretrained network and remove almost all plasticity. This is a practical solution, but almost entirely depends on the quality of the pretrained representation, without adding new information to the model. Beyond knowledge accumulation, better representations also should result in better continual learners. A better representation can make learning easier as features can be re-used by later tasks and thus do not have to be overwritten (Cha et al., 2021), reducing the risk of additional forgetting. With a few samples (*e.g.* a replay memory), a strong representation can also be used to quickly recover past information, previously referred to as 'fast remembering' (Davari et al., 2022; Hadsell et al., 2020). An important step in enabling knowledge accumulating is thus preventing forgetting of features learned during a task, as shown by the ensemble baseline.

## 5 Can feature forgetting be prevented?

Over the last years, many methods to alleviate forgetting have been proposed. In this section, we review examples of some of the most important families of methods and evaluate how they deal with feature forgetting and knowledge accumulation. Additionally, we test alternatives to the often used supervised cross-entropy loss in continual learning. The choice of algorithms is not driven by finding the best possible method, but we try to cover the most central ideas, in their simplest form. We test replay with a simple experience replay algorithm with 20 samples per class (ER), parameter regularization using MAS (Aljundi et al., 2018) and functional regularization with LwF (Li & Hoiem, 2017). To test whether our results also hold in different training regimes we also report results using self-supervised learning with BarlowTwins (Barlow) (Zbontar et al., 2021) and contrastive learning with a supervised contrastive loss (SupCon) (Khosla et al., 2020).

Figure 7 shows the forgetting for the tested methods, Table 2 their learning accuracy, or how well they learn new tasks. Replay, MAS and LwF forget at least as much in the output as on their



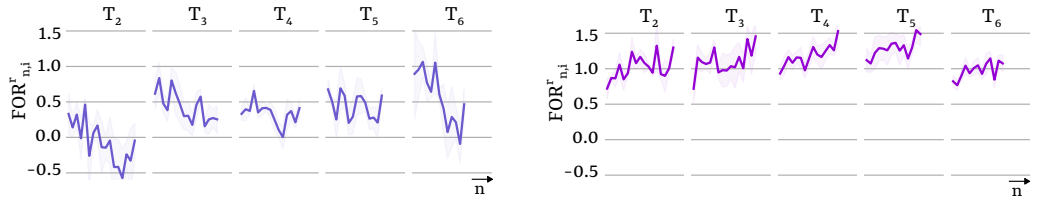

(a) Replay – ER (20 samples / class)

(b) Parameter regularization – MAS

(c) Functional regularization – LwF

(d) Ensemble of Section 4

Figure 7: Forgetting as in Equation 1, for $T_2$ to $T_6$ of the tested methods on the Mini-ImageNet sequence. (Mean $\pm$ standard error)




(a) Selfsupervised learning – Barlow

(b) Contrastive learning – SupCon

Figure 8: Forgetting for the first 5 tasks using selfsupervised and contrastive losses, instead of the default cross-entropy. (Mean $\pm$ standard error)

representation. LwF prevents a lot of forgetting, both on the representation and at the output level, although it does not close the gap with the ensemble - indicating that it still forgets. Replay and MAS have less forgetting than finetuning, but do not accumulate more knowledge. This is likely the results of their lower learning accuracy, they learn less, so they can accumulate less knowledge. The ensemble shows negative forgetting, because transfer from later tasks further improve performance. Figure 8 reports the other losses. The Barlow Twins baseline has a very 'noisy' forgetting curve. This is likely because the increase in performance during a task is rather small as it does not use any supervised information during training. Surprisingly SupCon forgets even more than it learned. Figure 6b shows the knowledge accumulated for each method, measured by the probing accuracy on a downstream task. Table 2 provides details on the overall improvement of the representations. Similar to the findings in Section 4, representation forgetting prevents knowledge accumulation, which remains true when using continual learning methods. LwF forgets least, and also builds up the most knowledge.

| | Finetune | Concat | Replay | MAS | LwF | Barlow | SupCon |
|---|---|---|---|---|---|---|---|
| $\overline{LP_{i,i}}$ | 86.0 $\pm$ 0.4 | 86.4 $\pm$ 0.4 | 81.4 $\pm$ 0.5 | 79.2 $\pm$ 0.5 | 85.1 $\pm$ 0.4 | 78.6 $\pm$ 0.6 | 85.2 $\pm$ 0.5 |
| $LP_{1,d}$ | 64.4 $\pm$ 3.8 | 64.5 $\pm$ 3.7 | 64.6 $\pm$ 3.9 | 64.6 $\pm$ 3.6 | 64.0 $\pm$ 3.3 | 65.6 $\pm$ 3.5 | 65.9 $\pm$ 3.7 |
| $LP_{19,d}$ | 70.4 $\pm$ 3.3 | 82.0 $\pm$ 1.7 | 70.2 $\pm$ 3.2 | 77.6 $\pm$ 2.9 | 71.5 $\pm$ 2.7 | 73.9 $\pm$ 1.8 | 64.1 $\pm$ 3.4 |
| $LP_{19,d}$–$LP_{1,d}$ | 6.0 $\pm$ 2.4 | 17.5 $\pm$2.7 | 5.6 $\pm$ 2.2 | 7.4 $\pm$1.2 | 13.0 $\pm$3.1 | 8.3 $\pm$1.8 | -1.8 $\pm$1.8 |

Table 2: Results of the representations of the different methods tested. $\overline{LP_{i,i}}$ denotes the average learning accuracy, *i.e.* the LP-accuracy of a task just after it was trained. The difference between final and initial LP-accuracy measures how much knowledge was accumulated during training. (Mean $\pm$ standard error)

## 6 DISCUSSION

**Representations forget catastrophically**. The results in Section 3 provide compelling evidence that when continually training a model, the information that was learned during a task is catastrophically forgotten. Moreover, in section 5 we find that for the various types of continual learning methods, the representation forgets as much as the observed performance. This seems contradictory to the claims from Davari et al. (2022), but they directly compared output and representation forgetting, not taking the baseline performance and knowledge accumulation into account.

**Forgetting and representation learning are part of the same problem**. In Section 4, we show that a model that is not subject to forgetting, yet learns in the same way as continual finetuning, has the best representation for unseen tasks, with all the discussed benefits. This is again confirmed in Section 5 where especially functional regularization (*e.g*. LwF) has a lot less forgetting, which is reflected by its stronger representation for downstream tasks (see Table 2). For other methods the representations do not significantly improve, although they have lower forgetting. This can be explained by their reduced learning capacity, *i.e*. the accuracy of a newly learned task is less high. This means less is learned, so with the same amount of forgetting there is less knowledge accumulation. See Supplemental for a further details on the learning capacity.

**Role of data and tasks**. As with every machine learning problem, also in this paper there might be a strong dependency on the used data. We tried to reduce this by evaluating all our findings on two datasets (Mini-Imagenet in the main paper, Cifar100 in Supplemental). When comparing performance during training, any metric is always measured on the *same* subset of classes, regardless of the training stage. Some recent works compare results on increasing large sets of classes, which confounds the comparison, as more classes makes for a more difficult problem Cha et al. (2022). Of course, this does not cover all cases. Most importantly, both datasets we used consist of natural images and the trained tasks belong to the same dataset. This makes the opportunity for knowledge transfer arguably larger than when using completely different datasets (not necessarily restricted to natural images). We leave this study for future work, yet hypothesize that with less knowledge accumulation, there is likely even more forgetting, as discussed in Section 3.

**Knowledge accumulation and feature forgetting**. In Section 3 we alluded on the difference between early and later tasks, and how the early tasks seemingly forget less. In Figure 6a we show how the finetune baseline accumulates knowledge and improves on a downstream task. Knowledge accumulation is stronger during the earlier tasks, although it is not a stark difference. Yet it might explain why earlier tasks forget less according to our measure: it is compensated by more knowledge accumulation.

**Future work**. We identify evaluating the current state of the art methods in light of our findings as future work, as well as an analysis on benchmarks that have less related tasks. Combining the benefits of functional regularization with strategies to remove biases in the head can be further investigated to combine the best of both worlds. Finally, as others have reported before, self-supervised and contrastive losses are a promising direction for continual learning (Cha et al., 2021; Davari et al., 2022), yet we showed that these approaches also suffer from feature forgetting.

## 7 RELATED WORK

**Representation learning**. Data rarely come in a format that is adapted to the task we want to perform (Bengio et al., 2013). Except for very simple problems, it is near impossible to directly classify images in their raw pixel representation. For example, many changes in the pixels (*e.g*. translation, rotation, illumination) do not alter the semantics of the image so they should not change the representation. For a long time, researchers have been searching for a representation of images that makes it convenient to solve semantic tasks. Handcrafting features was the standard, *e.g*. (Csurka et al., 2004), but this requires expert knowledge engineering and may not result in optimal features. Since the rise of deep learning, features are more commonly learned by neural networks, directly from the raw data. Both Bengio et al. (2013) and Goodfellow et al. (2016) define *good* representations as ones that make it easier to solve tasks of interest, a definition we adopt. They see deep neural networks as inevitable representation learners, even when this is not explicitly the goal. Neural networks trained to predict image-label pairs indirectly learn a representation where semantically different images are linearly separable in the output of the penultimate layer. Yet representations can

also be learned directly, which can improve robustness, boost generalization, or reduce the need for labeled data (Jing & Tian, 2020).

**Head vs. representation**. The paper proposing iCaRL (Rebuffi et al., 2017) is one of the first continual learning works to explicitly disentangle the representation and head. The head of a model can be relatively well learned with small subsets of data, *e.g*. in the case of classification as a linear layer or with non-parametric approaches like k-nearest neighbors (Wang et al., 2020; Taunk et al., 2019). On the other hand, heads do not transfer well, but quickly become disconnected from the representation when the representation changes while the head is static (Caccia et al., 2021). In the context of continual learning this property has been identified to impact performance severely, and methods updating the last layer only on small memories with balanced data, have shown successes in overcoming much of the observed forgetting (Wu et al., 2019; Zhao et al., 2020).

Recently some continual learning methods explicitly try to foster transfer of knowledge by taking inspiration from advances in representation learning (Jing & Tian, 2020). Some approaches apply contrastive losses (Cha et al., 2021; Mai et al., 2021) and self supervised learning (Marsocci & Scardapane, 2022; Hu et al., 2022; Fini et al., 2022; Rao et al., 2019) to improve continual learning performance, other works take ideas from meta-learning (Javed & White, 2019; Caccia et al., 2020) to learn representations that can easily adapt to new tasks. Lastly, Pham et al. (2021) take inspiration from neuroscience and combine fast and slow learners, *i.e*. supervised and self-supervised modules, in one system.

**Evaluating representation quality**. Effectively leveraging generalization and transfer properties of deep representations is one thing, evaluating their quality is another. As pointed out above, measuring forgetting at the output (the head) of a neural network does not tell us everything about the internal state of a network. Studies that retrain the last layer (Xiong et al., 2019), or a set of deeper layers (Murata et al., 2020), with the earlier layers frozen, hint that representations of lower layers are still useful for seemingly forgotten tasks. However, rather than these layers remembering something specific to the observed tasks, other works interpret this as better generalizability of the lower layers (Ramasesh et al., 2021; Yosinski et al., 2014; Zeiler & Fergus, 2014). Early layers may not seem to forget as much, because their representations are so general that they are almost fully reusable for future tasks, while deeper layers successively encode information more specific to the observed data, that is prone to being overwritten by information of new task's data (Ramasesh et al., 2021).

Davari et al. (2022) and Kim & Han (2023) use linear probes to measure forgetting of the representation in the penultimate layer. Davari et al. (2022) conclude that forgetting is less catastrophic and contrary to (Ramasesh et al., 2021) find that no task-critical information is lost. In contrast Kim & Han (2023) attest severe forgetting in the representation for the set of mechanisms evaluated in both works. The most notable difference in their experimentation setup is that Kim & Han (2023) pre-trainined the model's representation on half the respective dataset in advance. Additionally, the model's ability to incorporate new knowledge (plasticity) is investigated, with the result that most recent continual learning approaches that prevent forgetting at the same time diminish plasticity as well. Zhang et al. (2022) test the performance of a downstream task, showing that finetuning accumulates some knowledge.

## 8 Conclusion

In this work we studied how deep neural networks learn and forget representations when continually trained on a sequence of image classification tasks. If forgetting is calculated as the proportion of newly learned knowledge that is forgotten, representations forget at least as much as 'at the output'. Forgetting these representations reduces how much knowledge a model accumulates, as exemplified by the *ensemble* baseline. We further showed that feature forgetting is also observed when training using self-supervised and contrastive losses. Finally, we compared the feature forgetting and knowledge accumulation of different types of continual learning methods, whereby we found that functional regularization can prevent a large portion of representation forgetting. We hope that with the work we present here, future continual learning solutions will be evaluated not only on their output performance, but also on their representation quality and how they prevent feature forgetting.

## REFERENCES

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

Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1798–1828, 2013.

Lucas Caccia, Rahaf Aljundi, Tinne Tuytelaars, Joelle Pineau, and Eugene Belilovsky. Reducing representation drift in online continual learning. *arXiv preprint arXiv:2104.05025*, 1(3), 2021.

Massimo Caccia, Pau Rodriguez, Oleksiy Ostapenko, Fabrice Normandin, Min Lin, Lucas Page-Caccia, Issam Hadj Laradji, Irina Rish, Alexandre Lacoste, David Vázquez, et al. Online fast adaptation and knowledge accumulation (osaka): a new approach to continual learning. *Advances in Neural Information Processing Systems*, 33:16532–16545, 2020.

Hyuntak Cha, Jaeho Lee, and Jinwoo Shin. Co2l: Contrastive continual learning. In *Proceedings of the IEEE/CVF International conference on computer vision*, pp. 9516–9525, 2021.

Sungmin Cha, Dongsub Shim, Hyunwoo Kim, Moontae Lee, Honglak Lee, and Taesup Moon. Is continual learning truly learning representations continually? *arXiv preprint arXiv:2206.08101*, 2022.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pp. 1597–1607. PMLR, 2020.

Gabriella Csurka, Christopher Dance, Lixin Fan, Jutta Willamowski, and Cédric Bray. Visual categorization with bags of keypoints. In *Workshop on statistical learning in computer vision, ECCV*, volume 1, pp. 1–2. Prague, 2004.

MohammadReza Davari, Nader Asadi, Sudhir Mudur, Rahaf Aljundi, and Eugene Belilovsky. Probing representation forgetting in supervised and unsupervised continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16712–16721, 2022.

Matthias De Lange, Rahaf Aljundi, Marc Masana, Sarah Parisot, Xu Jia, Aleš Leonardis, Gregory Slabaugh, and Tinne Tuytelaars. A continual learning survey: Defying forgetting in classification tasks. *IEEE transactions on pattern analysis and machine intelligence*, 44(7):3366–3385, 2022.

Enrico Fini, Victor G Turrisi Da Costa, Xavier Alameda-Pineda, Elisa Ricci, Karteek Alahari, and Julien Mairal. Self-supervised models are continual learners. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9621–9630, 2022.

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

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

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Dapeng Hu, Shipeng Yan, Qizhengqiu Lu, HONG Lanqing, Hailin Hu, Yifan Zhang, Zhenguo Li, Xinchao Wang, and Jiashi Feng. How well does self-supervised pre-training perform with streaming data? In *International Conference on Learning Representations*, 2022.

Paul Janson, Wenxuan Zhang, Rahaf Aljundi, and Mohamed Elhoseiny. A simple baseline that questions the use of pretrained-models in continual learning. *arXiv preprint arXiv:2210.04428*, 2022.

Khurram Javed and Martha White. Meta-learning representations for continual learning. *Advances in neural information processing systems*, 32, 2019.

Longlong Jing and Yingli Tian. Self-supervised visual feature learning with deep neural networks: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 43(11):4037–4058, 2020.

Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. Supervised contrastive learning. *Advances in neural information processing systems*, 33:18661–18673, 2020.

Dongwan Kim and Bohyung Han. On the stability-plasticity dilemma of class-incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 20196–20204, 2023.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. URL http://arxiv.org/abs/1412.6980.

Akinwande Komolafe. Retraining model during deployment: Continuous training and continuous testing, 2023. URL https://neptune.ai/blog/retraining-model-during-deployment-continuous-training-continuous-testing. Online; accessed 30-June-2023.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.

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

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

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

Zheda Mai, Ruiwen Li, Hyunwoo Kim, and Scott Sanner. Supervised contrastive replay: Revisiting the nearest class mean classifier in online class-incremental continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3589–3599, 2021.

Valerio Marsocci and Simone Scardapane. Continual barlow twins: continual self-supervised learning for remote sensing semantic segmentation. *arXiv preprint arXiv:2205.11319*, 2022.

Michael McCloskey and Neal J Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of learning and motivation*, volume 24, pp. 109–165. Elsevier, 1989.

Kengo Murata, Tetsuya Toyota, and Kouzou Ohara. What is happening inside a continual learning model? a representation-based evaluation of representational forgetting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pp. 234–235, 2020.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pp. 8024–8035. Curran Associates, Inc., 2019. URL http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.

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

Vinay V Ramasesh, Ethan Dyer, and Maithra Raghu. Anatomy of catastrophic forgetting: Hidden representations and task semantics. *ICLR 2020*, 2021.

Dushyant Rao, Francesco Visin, Andrei Rusu, Razvan Pascanu, Yee Whye Teh, and Raia Hadsell. Continual unsupervised representation learning. *Advances in Neural Information Processing Systems*, 32, 2019.

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

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

Anthony Robins. Catastrophic forgetting in neural networks: the role of rehearsal mechanisms. In *Proceedings 1993 The First New Zealand International Two-Stream Conference on Artificial Neural Networks and Expert Systems*, pp. 65–68. IEEE, 1993.

Kashvi Taunk, Sanjukta De, Srishti Verma, and Aleena Swetapadma. A brief review of nearest neighbor algorithm for learning and classification. In *2019 International Conference on Intelligent Computing and Control Systems (ICCS)*, pp. 1255–1260. IEEE, 2019.

Gido M van de Ven, Tinne Tuytelaars, and Andreas S Tolias. Three types of incremental learning. *Nature Machine Intelligence*, 4(12):1185–1197, 2022.

Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Daan Wierstra, et al. Matching networks for one shot learning. *Advances in neural information processing systems*, 29, 2016.

Joshua T Vogelstein, Jayanta Dey, Hayden S Helm, Will LeVine, Ronak D Mehta, Tyler M Tomita, Haoyin Xu, Ali Geisa, Qingyang Wang, Gido M van de Ven, et al. Representation ensembling for synergistic lifelong learning with quasilinear complexity. *arXiv preprint arXiv:2004.12908*, 2020.

Yaqing Wang, Quanming Yao, James T Kwok, and Lionel M Ni. Generalizing from a few examples: A survey on few-shot learning. *ACM computing surveys (csur)*, 53(3):1–34, 2020.

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

Yuwen Xiong, Mengye Ren, and Raquel Urtasun. Learning to remember from a multi-task teacher. *arXiv preprint arXiv:1910.04650*, 2019.

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

Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? *Advances in neural information processing systems*, 27, 2014.

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

Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part I 13*, pp. 818–833. Springer, 2014.

Xiao Zhang, Dejing Dou, and Ji Wu. Feature forgetting in continual representation learning. *arXiv preprint arXiv:2205.13359*, 2022.

Bowen Zhao, Xi Xiao, Guojun Gan, Bin Zhang, and Shu-Tao Xia. Maintaining discrimination and fairness in class incremental learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 13208–13217, 2020.

## SUPPLEMENTARY MATERIAL

The supplement material contains additional information on the implementation details and extra results for the experimentation in the main text.

## A EXPERIMENTATION DETAILS

This section details the training and evaluation of all experiments in the main paper and supplemental material, unless explicitly stated to deviate.

**Data** MiniImageNet consits of $50,000$ train and $10,000$ test RGB-images of resolution $84 \times 84$ equally divided over $100$ classes. We split this dataset into 20 disjoint tasks such that each task contains five classes. The second benchmark is Split CIFAR-100, which is based on the CIFAR-100 dataset (Krizhevsky et al., 2009) with the same amount of RGB-images and classes as MiniImageNet, but with reduced resolution of $32 \times 32$. We split this dataset into ten disjoint tasks with ten classes each. All experiments are run with five different seeds that also shuffle the class splits over the tasks. See Table 3 and Table 4 for the exact sequences.

**Architecture and optimization** Throughout this work ResNet-18 (He et al., 2016) is the base architecture for all models. For MiniImageNet we adopt the implementation as default in the pytorch-torchvision (Paszke et al., 2019) library. For CIFAR-100 we employed the slim version of the model as proposed by Lopez-Paz & Ranzato (2017). All networks are trained from scratch, and pre-trained networks are considered future work. The optimization schedules are adjusted with respect to the training criterion. For supervised training with the cross-entropy loss we use an AdamW (Loshchilov & Hutter, 2017) optimizer with static learning rate of $0.001$, weight decay $0.0005$, and beta-values $0.9$ and $0.999$. Each task is trained for $50$ epochs with mini-batches of size $128$.

For the SupCon (Khosla et al., 2020) and BarlowTwins (Zbontar et al., 2021) optimization criteria, we stuck to optimization schedules proposed in literature for their application to continual learning. In line with observations by Cha et al. (2021), the SupCon training regime uses an SGD optimizer with momentum $0.9$. The learning rate is scheduled in the same way for every task warming up from $0.0005$ to $0.1$ in the first ten epochs, then annealing by a cosine schedule back to its starting value. The first task is trained for $500$ epochs, all subsequent tasks for $100$ epochs, with a batch size of $256$. The projection network necessary for this objective consists of an MLP with (single) hidden dimension of $512$, projecting to a $128$ dimensional space. Barlow-Twins optimization is aligned to (Marsocci & Scardapane, 2022; Fini et al., 2022). We use an Adam optimizer (Kingma & Ba, 2015) with learning rate $0.0001$ and weight decay $0.0005$. We train $500$ epochs for each task with batch size of $256$. Again, the projection head is an MLP but with two hidden layers, and hidden and final projection dimension of $2048$. All methods use the same augmentations, see below.

**Probe optimization** To quantify the quality of the representation we apply probes based on linear- and k-nearest neighbors- (*k*NN) classifiers. Linear classifiers consist of a single linear layer. In the optimal probing case, reported mostly throughout the work, it is optimized with access to all training data. Linear probes are optimized analog to Cha et al. (2021). Keeping a batch-size of 128, we use SGD with momentum of $0.9$ and no weight decay for 100 epochs. The learning rate of $0.1$ is decaying at epochs 60, 75, and 90 by a factor of $0.2$. Similarly, *k*NN uses all training data to evaluate the representations.

**Continual learning mechanisms** LwF and MAS are using a value of $\lambda = 1.0$ as advocated by its original authors. Replay uses a random selection of $20$ exemplars per class. The weight of the loss on replayed samples is increased proportionally to the number of previously observed tasks, to prevent favoring the current task in the optimization. An upper bound is reported by jointly training the model on all observed data. For our lower-bound we want to document the impact the singled out tasks have. This we achieve by re-initializing the model before training a new task, but allowing the new task to train for as many iterations as a continual model would have, *e.g.* 50 epochs for the first task, then 100 for the second, and so on. By design this model has zero transfer of knowledge, and we will refer to it as 'Single task' baseline.

**Augmentations** In all experiments we use the data augmentation pipeline from SimCLR Chen et al. (2020). The augmentations pipeline consists of random crops and horizontal flips, color-jitter (brightness=0.4, contrast=0.4, saturation=0.2, hue=0.1), random grayscaling (p=20%) and Gaussian blur using a kernel of size 9 and sigma range 0.1 to 0.2. In PyTorch, the augmentations are defined as follows:

```
from torchvision.transforms import *

RandomHorizontalFlip(p=0.5),
RandomResizedCrop(size=(32, 32), scale=(0.2, 1.0)),
RandomApply(
    [ColorJitter(brightness=0.4, contrast=0.4, saturation=0.2, hue=0.1)], p=0.8),
RandomGrayscale(p=0.2),
RandomApply([
    GaussianBlur(kernel_size=input_size[0]//20*2+1, sigma=(0.1, 2.0))], p=0.5)
```

## B RELATIVE FORGETTING: EXTRA RESULTS

Figure 3 only shows the relative forgetting in the Mini-ImageNet task sequence. For completness, we report here the relative forgetting of all tasks.

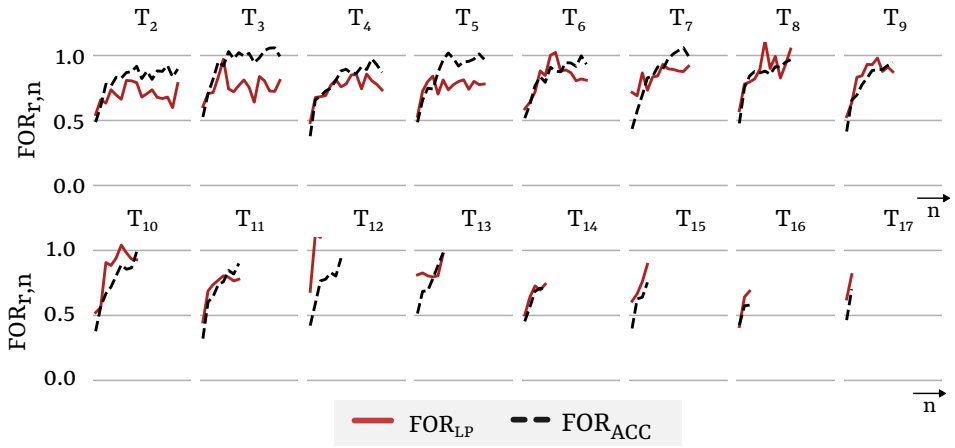



Figure 9: Representation and observed forgetting using linear probes for all tasks in Mini-ImageNet using finetuning (except first and last, for which we cannot calculate relative forgetting)

## C ENSEMBLE: FURTHER DETAILS

The ensemble method trains stores a model copy every after every task. Each of these models output a representation $f_i$ with dimension $k$ for the input data. During training, only the model of the task is used and the others ae frozen. Before evaluating and training of the linear probes during evaluation, all of the representations $f_t$ are concatenated to form one large representation $f = [f_1, f_2 \cdots f_t]$. On top of this large representation a linear layer with input dimension $tk$ is trained, instead of just $k$ for the finetuned model.

To mitigate the influence of the higher dimension, for which it might be easier to find linearly separable features, we add a dimension reduction to lower the dimension back to $k$. We do this by projecting the features of the ensemble on the top-$k$ most significant PCA dimensions. The results are shown in Figure 10. The reduced ensemble performs a bit worse than the full ensemble, yet still significantly better than finetuning.

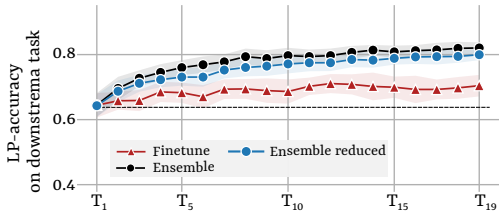



Figure 10: LP-accuracies of finetuning, the ensemble baseline and its reduced version, as explain in Section C

## D RESULTS ON CIFAR100

To reduce the dependency on only having experiments on a single dataset, we report our main results here also on CIFAR100. The results on CIFAR100 follow the same general trends as those on Mini-ImageNet in the main paper. The largest difference is that the effects are sometimes smaller, due to the shorter task sequence.

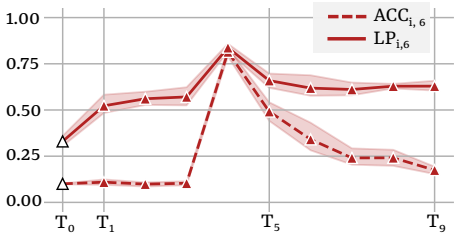



Figure 11: Linear probe and output accuracy of $T_3$ during the entire CIFAR100 sequence.

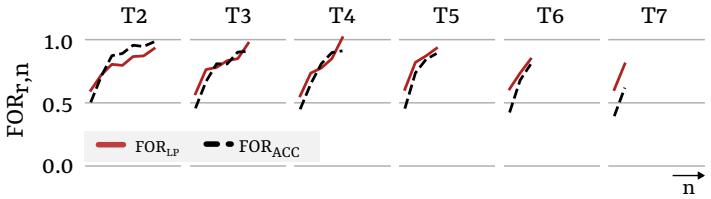



Figure 12: Representation and observed forgetting using linear probes for all tasks in CIFAR100 using finetuning (except first and last task, for which we cannot calculate relative forgetting)

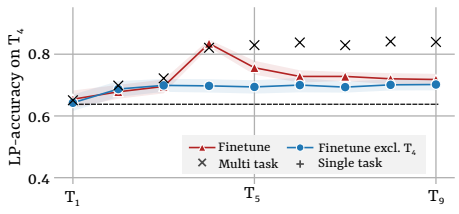



Figure 13: Finetune, exclusion, single task and multi task with CIFAR100.

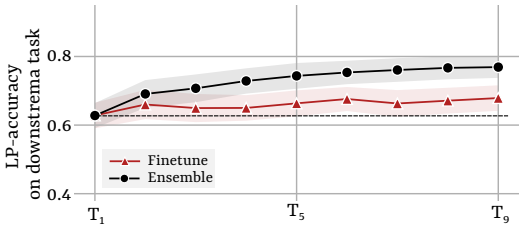



Figure 14: Comparing the ensemble and finetuning on CIFAR100.

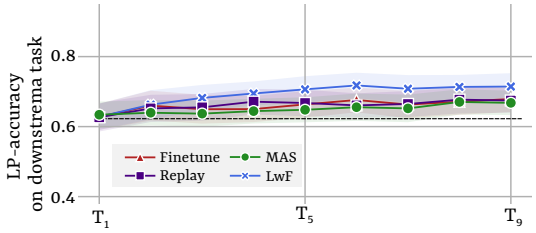



Figure 15: LP-accuracies on a downstream task of CIFAR100.

## E EVALUATION WITH *k*NN

In this section we report the most important results of the main paper using *k*NN instead of using linear probes. This has the benefit that it there are no hyperparameters to tune and does not depend on the optimization used. We report it here for completness, and keep the linear probes in the main paper as this is how preivous papers reported their results Davari et al. (2022); Cha et al. (2022); Zhang et al. (2022). In general, the results in Figure 16, 17, 18 and 19 follow the same trends as observed in the main paper, with the main difference that the absolute values are lower, likely due to the suboptimalitiy of *k*NN compared to linear probe optimization.

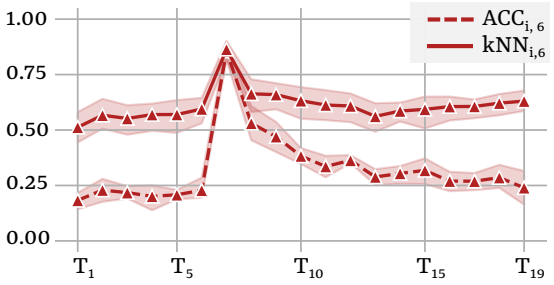



Figure 16: *k*NN and output accuracy of $T_6$ during the entire Mini-ImageNet sequence.

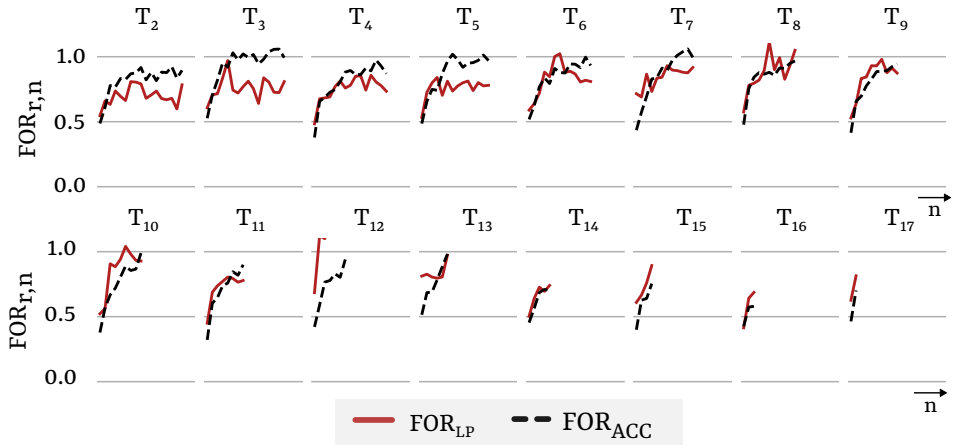



Figure 17: Representation and observed forgetting using *k*NN for all tasks in Mini-ImageNet (except last and first, for which we cannot calculate relative forgetting)

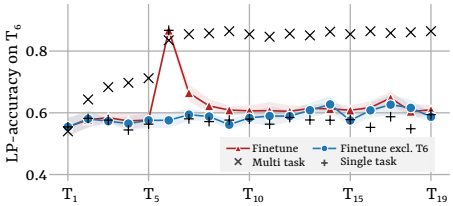



Figure 18: Finetune, exclusion, single task and multi task using *k*NN.

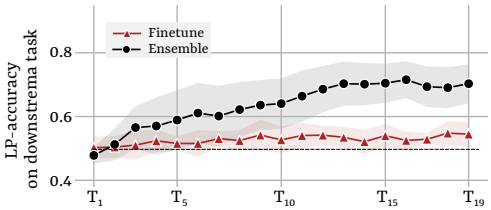



Figure 19: Comparing the ensemble and finetuning using *k*NN.

## F DETAILED TASK SEQUENCE INFORMATION

In Table 3 and Table 4 we report the exact task sequences used in the experiments in the main paper. These are chosen at random, but consistent in all experiments. The randomness of the tasks also means that there difficult varies quite a bit, which explais some of the higher standard errors in the experiments reported.

| idx | Synset | Synset name | idx | Synset | Synset name | idx | Synset | Synset name |
|---|---|---|---|---|---|---|---|---|
| 0 | n01532829 | house_finch | 33 | n03400231 | frying_pan | 66 | n02981792 | catamaran |
| 1 | n01558993 | robin | 34 | n03476684 | hair_slide | 67 | n03980874 | poncho |
| 2 | n01704323 | triceratops | 35 | n03527444 | holster | 68 | n03770439 | miniskirt |
| 3 | n01749939 | green_mamba | 36 | n03676483 | lipstick | 69 | n02091244 | Ibizan_hound |
| 4 | n01770081 | harvestman | 37 | n03838899 | oboe | 70 | n02114548 | white_wolf |
| 5 | n01843383 | toucan | 38 | n03854065 | organ | 71 | n02174001 | rhinoceros_beetle |
| 6 | n01910747 | jellyfish | 39 | n03888605 | parallel_bars | 72 | n03417042 | garbage_truck |
| 7 | n02074367 | dugong | 40 | n03908618 | pencil_box | 73 | n02971356 | carton |
| 8 | n02089867 | Walker_hound | 41 | n03924679 | photocopier | 74 | n03584254 | iPod |
| 9 | n02091831 | Saluki | 42 | n03998194 | prayer_rug | 75 | n02138441 | meerkat |
| 10 | n02101006 | Gordon_setter | 43 | n04067472 | reel | 76 | n03773504 | missile |
| 11 | n02105505 | komondor | 44 | n04243546 | slot | 77 | n02950826 | cannon |
| 12 | n02108089 | boxer | 45 | n04251144 | snorkel | 78 | n01855672 | goose |
| 13 | n02108551 | Tibetan_mastiff | 46 | n04258138 | solar_dish | 79 | n09256479 | coral_reef |
| 14 | n02108915 | French_bulldog | 47 | n04275548 | spider_web | 80 | n02110341 | dalmatian |
| 15 | n02111277 | Newfoundland | 48 | n04296562 | stage | 81 | n01930112 | nematode |
| 16 | n02113712 | miniature_poodle | 49 | n04389033 | tank | 82 | n02219486 | ant |
| 17 | n02120079 | Arctic_fox | 50 | n04435653 | tile_roof | 83 | n02443484 | black-footed_ferret |
| 18 | n02165456 | ladybug | 51 | n04443257 | tobacco_shop | 84 | n01981276 | king_crab |
| 19 | n02457408 | three-toed_sloth | 52 | n04509417 | unicycle | 85 | n02129165 | lion |
| 20 | n02606052 | rock_beauty | 53 | n04515003 | upright | 86 | n04522168 | vase |
| 21 | n02687172 | aircraft_carrier | 54 | n04596742 | wok | 87 | n02099601 | golden_retriever |
| 22 | n02747177 | ashcan | 55 | n04604644 | worm_fence | 88 | n03775546 | mixing_bowl |
| 23 | n02795169 | barrel | 56 | n04612504 | yawl | 89 | n02110063 | malamute |
| 24 | n02823428 | beer_bottle | 57 | n06794110 | street_sign | 90 | n02116738 | African_hunting_dog |
| 25 | n02966193 | carousel | 58 | n07584110 | consomme | 91 | n03146219 | cuirass |
| 26 | n03017168 | chime | 59 | n07697537 | hotdog | 92 | n02871525 | bookshop |
| 27 | n03047690 | clog | 60 | n07747607 | orange | 93 | n03127925 | crate |
| 28 | n03062245 | cocktail_shaker | 61 | n09246464 | cliff | 94 | n03544143 | hourglass |
| 29 | n03207743 | dishrag | 62 | n13054560 | bolete | 95 | n03272010 | electric_guitar |
| 30 | n03220513 | dome | 63 | n13133613 | ear | 96 | n07613480 | trifle |
| 31 | n03337140 | file | 64 | n03535780 | horizontal_bar | 97 | n04146614 | school_bus |
| 32 | n03347037 | fire_screen | 65 | n03075370 | combination_lock | 98 | n04418357 | theater_curtain |

Table 3: The classes included in Split MiniImagenet, with their index, (which is not general, but used in the task splits), their synsets and their name.

| | Seed 42 | Seed 52 | Seed 62 | Seed 72 | Seed 82 |
|---|---|---|---|---|---|
| T1 | 83 - 53 - 70 - 45 - 44 | 82 - 8 - 44 - 19 - 2 | 76 - 48 - 62 - 80 - 29 | 76 - 82 - 43 - 16 - 84 | 72 - 33 - 58 - 2 - 55 |
| T2 | 39 - 22 - 80 - 10 - 0 | 73 - 37 - 89 - 67 - 18 | 99 - 60 - 89 - 39 - 69 | 95 - 78 - 91 - 30 - 22 | 84 - 54 - 75 - 28 - 40 |
| T3 | 18 - 30 - 73 - 33 - 90 | 4 - 92 - 83 - 24 - 14 | 14 - 74 - 59 - 87 - 55 | 1 - 96 - 25 - 81 - 62 | 39 - 15 - 41 - 12 - 35 |
| T4 | 4 - 76 - 77 - 12 - 31 | 93 - 90 - 84 - 81 - 66 | 40 - 46 - 54 - 92 - 7 | 5 - 18 - 63 - 14 - 24 | 23 - 49 - 91 - 32 - 38 |
| T5 | 55 - 88 - 26 - 42 - 69 | 40 - 72 - 56 - 36 - 51 | 6 - 32 - 77 - 27 - 63 | 23 - 75 - 9 - 60 - 27 | 64 - 68 - 6 - 92 - 18 |
| T6 | 15 - 40 - 96 - 9 - 72 | 50 - 68 - 88 - 55 - 57 | 96 - 33 - 49 - 25 - 68 | 83 - 20 - 90 - 55 - 36 | 48 - 47 - 13 - 89 - 79 |
| T7 | 11 - 47 - 85 - 28 - 93 | 27 - 29 - 80 - 3 - 94 | 26 - 94 - 38 - 85 - 98 | 4 - 10 - 77 - 93 - 33 | 96 - 22 - 34 - 81 - 63 |
| T8 | 5 - 66 - 65 - 35 - 16 | 53 - 62 - 87 - 52 - 95 | 61 - 43 - 93 - 15 - 28 | 58 - 35 - 97 - 11 - 59 | 53 - 85 - 14 - 50 - 44 |
| T9 | 49 - 34 - 7 - 95 - 27 | 70 - 12 - 1 - 97 - 48 | 36 - 2 - 42 - 75 - 31 | 56 - 98 - 47 - 86 - 38 | 24 - 61 - 11 - 0 - 21 |
| T10 | 19 - 81 - 25 - 62 - 13 | 60 - 47 - 65 - 10 - 41 | 22 - 56 - 3 - 67 - 19 | 85 - 66 - 49 - 41 - 87 | 10 - 59 - 90 - 71 - 56 |
| T11 | 24 - 3 - 17 - 38 - 8 | 17 - 96 - 9 - 49 - 30 | 20 - 90 - 50 - 84 - 66 | 42 - 99 - 57 - 0 - 6 | 17 - 76 - 1 - 95 - 70 |
| T12 | 78 - 6 - 64 - 36 - 89 | 38 - 58 - 0 - 26 - 21 | 70 - 97 - 4 - 64 - 44 | 70 - 13 - 50 - 40 - 68 | 94 - 37 - 5 - 4 - 26 |
| T13 | 56 - 99 - 54 - 43 - 50 | 31 - 15 - 75 - 25 - 6 | 82 - 47 - 95 - 41 - 51 | 48 - 73 - 37 - 8 - 39 | 60 - 20 - 45 - 98 - 74 |
| T14 | 67 - 46 - 68 - 61 - 97 | 74 - 59 - 64 - 43 - 34 | 23 - 5 - 79 - 88 - 34 | 32 - 3 - 89 - 51 - 44 | 62 - 57 - 73 - 97 - 87 |
| T15 | 79 - 41 - 58 - 48 - 98 | 20 - 77 - 7 - 78 - 71 | 16 - 35 - 52 - 71 - 72 | 17 - 54 - 15 - 67 - 2 | 46 - 51 - 7 - 82 - 83 |
| T16 | 57 - 75 - 32 - 94 - 59 | 22 - 39 - 63 - 76 - 85 | 57 - 12 - 1 - 13 - 86 | 31 - 52 - 61 - 34 - 71 | 19 - 88 - 9 - 8 - 52 |
| T17 | 63 - 84 - 37 - 29 - 1 | 79 - 45 - 61 - 42 - 46 | 78 - 8 - 21 - 91 - 83 | 64 - 92 - 65 - 53 - 28 | 30 - 65 - 16 - 36 - 69 |
| T18 | 52 - 21 - 2 - 23 - 87 | 54 - 91 - 16 - 5 - 33 | 10 - 0 - 65 - 73 - 37 | 72 - 80 - 12 - 45 - 21 | 25 - 67 - 43 - 29 - 42 |
| T19 | 91 - 74 - 86 - 82 - 20 | 35 - 98 - 69 - 32 - 99 | 45 - 30 - 17 - 53 - 58 | 29 - 7 - 26 - 79 - 69 | 78 - 80 - 31 - 86 - 93 |
| Downstream task | 60 - 71 - 14 - 92 - 51 | 86 - 23 - 13 - 11 - 28 | 11 - 9 - 81 - 24 - 18 | 94 - 74 - 46 - 19 - 88 | 77 - 27 - 99 - 66 - 3 |

Table 4: Task splits used in the results with Split MiniImagenet. The indices correspond to the classes listed in Table 3. Results reported on Split MiniImagenet average over these 5, randomly determined, task sequences.