Momentum-based Weight Interpolation of Strong Zero-Shot Models for Continual Learning

Zafir Stojanovski1,∗, Karsten Roth1,∗, Zeynep Akata1,2
1University of Tübingen, 2MPI for Intelligent Systems

Abstract

Large pre-trained, zero-shot capable models have shown considerable success both for standard transfer and adaptation tasks, with particular robustness towards distribution shifts. In addition, subsequent fine-tuning can considerably improve performance on a selected downstream task. However, through naive fine-tuning, these zero-shot models lose their generalizability and robustness towards distribution shifts. This is a particular problem for tasks such as Continual Learning (CL), where continuous adaptation has to be performed as new task distributions are introduced sequentially. In this work, we showcase that where fine-tuning falls short to adapt such zero-shot capable models, simple momentum-based weight interpolation can provide consistent improvements for CL tasks in both memory-free and memory-based settings. In particular, we find improvements of over +4% on standard CL benchmarks, while reducing the error to the upper limit of jointly training on all tasks at once by more than half, allowing the continual learner to inch closer to the joint training limits.

1 Introduction

Continual Learning (CL) tackles the problem of learning from a non-stationary data stream, where training data is presented to the model not at once, but only in a sequence, and with limited capacity for retention and retraining. Not only does this require effective use of previously seen data, but also adaptation to novel context under continuously changing distribution shifts without catastrophic forgetting [11, 33, 35, 36]. Use cases are widespread, ranging from particularly compute-, time- or memory-limited to privacy-concerned applications [14, 3, 16, 40, 25].

Consequentially, previous research has introduced a wide range of methods to address training under continual shifts, such as through the use of efficient data replay [7, 8, 1, 28], regularization on the training dynamics [11, 35] or optimization procedures seeking for flat minima [23, 53]. Generally, these methods start from an untrained model which is then adapted to the data stream at hand. While this has found practical success, more recently the use of large-scale pre-trained models (“foundation models” [2, 25]) has become ubiquitous, as they have shown strong zero-shot generalizability to a variety of downstream tasks, with strong robustness to distribution shifts [2].

Their application to the CL problem set, which tackles a continuous distribution shift, stands to reason, with recent works showing notable benefits in the use of foundation models [40, 21, 31, 42], particularly highlighting a reduction in catastrophic forgetting. Still, as learners are adapted to continuously shifting training distribution, even foundation models will suffer from forgetting through fine-tuning [41].

∗Denotes equal contribution.

To maximize the benefits we can extract from the main continual learning process as well as the ability to classify novel samples at test time, it is thus important to minimize the impact on the generalizability of the adapted foundation model in order to account for potential further adaptations. To allow for improved deployment in the CL setting, in this work we show how momentum-based weight-interpolation can help remedy issues of such models adapted in a continual fashion. In particular, as we want to maximally retain the generalizability of our adapted foundation model, we introduce a bifurcated adaptation mechanism by retaining an additional copy of the initial foundation model (denoted as slow model). This slow model is excluded from the direct CL optimization process, and is only updated through linear momentum-interpolation with a task-adapted model copy (denoted as fast model).

This is motivated by insights made in [41], who show that simple linear interpolation in weight space between the original zero-shot model and a variant fine-tuned to a task at hand allows for adaptation, while retaining better generalizability as compared to sole fine-tuning. However, retaining a large collection of fine-tuned task expert models in the CL setting is memory intensive, impractical, and undesired. Instead, we show that we can simulate the empirical benefits highlighted in [41] through repeated momentum interpolation between our foundation model and a continuously fine-tuned variant. This allows us to avoid the drawbacks of pure fine-tuning, while both specializing on the new stream of tasks, and retaining the generalizability of our foundation model.

Indeed, experiments on three standard CL benchmarks (Seq-CIFAR-10, Seq-CIFAR-100 and Seq-Tiny-ImageNet) show improvements in class- and task-incremental settings on both memory-based and memory-free methods by up to +4%, and partly more than halving the error to the joint training performance bound. These results indicate that for practical usage of foundation models in a continuously distribution-shifting training scenario, momentum-based weight interpolation can be a reliable tool for consistent improvements that works well alongside any CL method.

2 Related Work

Regularization-based methods augment the training objective to mitigate forgetting by keeping the current parameters close to previous task parameters, such as through moment matching [15] or Elastic Weight Consolidation (EWC) [12], which performs Laplace approximations on the parameter posterior for each preceding task, using the means and covariances to regularize the current parameters via Mahalanobis distance minimization. Online Elastic Weight Consolidation (oEWC) [36] computes a momentum average of a single covariance matrix, and keeps the parameters from the last task only. Learning without Forgetting (LwF) [17] also keeps the parameters from the last task and adds a cross-entropy term between logits computed with the old and current parameters, using data from the current task. [22] show that dropout forces the model to learn a gating such that for different tasks, different paths of the network are active.

Rehearsal-based methods utilize Experience Replay [32] [33] by storing a small subset of the training data into a buffer, and continually replaying it as the model moves on to learn new tasks. Dark Experience Replay (DER) [4] introduces regularization in the rehearsal scheme by matching the logits of the past with the logits computed by the current network parameters. Gradient Episodic Memory (GEM) [18] and Average Gradient Episodic Memory (A-GEM) [8] enforce optimization constraints in the current task using data from past tasks. GDumb [29] greedily stores samples in memory, and only trains the model at test time using buffer data. DualNet [27] uses a slow network for learning task-agnostic features through Self-Supervised Learning, and a fast network for learning task-specific features. Contrastive Continual Learning (Co2L) [5] learns contrastive task-agnostic features, and trains a linear classifier using only buffer data. Our approach also bears conceptual similarities to the lookahead-style of optimization (see e.g. [43]), adapted to the continual learning problem.

Flatness-seeking methods aim to operate in flat minima regions for each task in sequence, thereby retaining antecedent performance. Finding Flat Minima (F2M) [59] independently adds small random noise to the parameters, thereby obtaining similar but different loss functions which are optimized jointly in order to locate flat minima. [24] studies how batch size, dropout and learning rate decay
affect the model’s ability to find flat basins. \cite{20} uses the Sharpness-Aware Minimization (SAM) \cite{9} procedure, which explicitly optimizes for parameters lying in flat basins.

3 Method

In CL, a model $f_\theta$ is trained on a sequence of $T$ tasks, where for each task $t \in \{1, \ldots, T\}$ the learner only gets access to a subset of samples $D_t = \{(x_i, y_i)\}_{i=1}^{N_t}$, but is eventually evaluated on joint performance, i.e. we optimize

$$\theta^* = \arg \min_{\theta} \sum_{t=1}^{T} \mathbb{E}_{(x,y) \sim D_t} [L(f_\theta(x), y)].$$

The main challenge is that at task $t$, the model has no access to data from previous tasks $\tilde{t} \in \{1, \ldots, t - 1\}$, therefore violating the typical IID data assumption. In this work, we investigate both the class-incremental setting, where subsets of classes are introduced in sequence, and the much easier task-incremental setting which jointly also provides respective task ids.

3.1 Momentum-based Weight Interpolation for Continual Learning (MCL)

To allow for effective and continuous adaptation of foundation models, we introduce momentum-based weight interpolation for CL. As our primary target is the retention of the generalizability and shift robustness of the underlying foundation model, it is important that minimal adaptation and fine-tuning is performed, while still allowing for a certain degree of adaptation to the target tasks at hand. For that, we suggest a retention of a slow model copy $\theta_{\text{slow}}$ which is kept disconnected from the entire adaptation process, while a second instantiation $\theta_{\text{fast}}$ is updated throughout the continual learning process. As $\theta_{\text{fast}}$ adapts to the target distribution at hand, at every iteration we simultaneously perform an iterative updating on our slow weights through weight-space interpolation:

$$\theta_{\text{slow}} = \tau \cdot \theta_{\text{slow}} + (1 - \tau) \cdot \theta_{\text{fast}}$$

where $\tau$ is our momentum hyperparameter. A simplified version of the procedure is summarized in Algorithm 1. As this mechanism is task- and memory-agnostic with no dependence on task boundaries, it can be applied to any continual learning framework, both memory-based and memory-free. And while straightforward and simple, the intuitively better retention of foundation model weights in the continual learning setting is well motivated.

Beyond a conceptual connection to the Complimentary Learning Systems (CLS) \cite{19, 6} theory from neuroscience which depicts human continual learning as an interplay of a fast adaptive and a slow retentive system, on a methodological level \cite{10} show that maintaining a running average of weights leads to wider optima and retained generalization during the standard fine-tuning process of a pre-trained model.

In addition, \cite{41} showcase that zero-shot and fine-tuned model weights are often connected by a linear path which retains performance. It therefore stands to reason that our linear momentum-based interpolation across task iterations allows us to connect to the performance of our task-adapted fast variant, while maintaining the generalizability our foundation model weights $\theta_{\text{slow}}$. The consequently sustained implicit optimization for a flatter minimum around $\theta_{\text{slow}}$, which is only updated through momentum-based interpolation, has strong ties to improved generalization across task sequences in continual learning \cite{39, 24, 9}, which we see reflected in our benchmark experiments in the next section.

4 Experiments

Datasets. We evaluate our method on three datasets commonly used in the literature: CIFAR-10 \cite{13}, CIFAR-100 \cite{13}, and Tiny ImageNet. We split each dataset into several tasks of non-overlapping classes: Seq-CIFAR-10 consisting of 5 tasks (2 classes each) and Seq-CIFAR-100/Seq-Tiny-ImageNet consisting of 10 tasks (10 and 20 classes each, respectively).
Algorithm 1: Momentum-based Weight Interpolation for Continual Learning (MCL)

Require: Pre-trained weights \( \theta_{\text{pre}} \), Momentum \( \tau \in [0, 1] \)

1: \( \theta_{\text{fast}} \leftarrow \theta_{\text{pre}} \)
2: \( \theta_{\text{slow}} \leftarrow \theta_{\text{pre}} \)
3: for \( t \leftarrow 1 \ldots \text{num\_tasks} \) do
   4:     for \( e \leftarrow 1 \ldots \text{num\_epochs} \) do
      5:         for (\( x, y \)) \sim D_t \) do
         6:             \( \theta_{\text{fast}} \leftarrow \theta_{\text{fast}} - \alpha \nabla \mathcal{L}(f_{\theta_{\text{fast}}}(x), y) \)
            \( \theta_{\text{slow}} \leftarrow \tau \cdot \theta_{\text{slow}} + (1 - \tau) \cdot \theta_{\text{fast}} \)
      7:     end for
   8: end for

Training. For our zero-shot model we use a pre-trained CLIP ViT-B/16 [30]. We built our CL experiments on [4] which implements several CL benchmarks in PyTorch [26]. All methods follow a standardized training protocol - trained on Nvidia 2080Ti’s using SGD [37], a fixed learning rate and no scheduler, with the same fine-tuning budget of 10 epochs. We perform grid searches on a random train subset to select the best learning rate \( \alpha \in \{10^{-2}, 10^{-3}, 10^{-4}, 10^{-5}, 10^{-6}, 10^{-7}\} \) as well as the best momentum strength \( \tau \in \{0.995, 0.997, 0.999, 0.9995, 0.9997, 0.9999\} \). We refer the reader to the appendix (§A.1) for an ablation study of the hyperparameters.

Evaluation. For both Task Incremental Learning (Task-IL) and Class Incremental Learning (Class-IL) scenarios, we report the final classification accuracy over all encountered classes, with task identities also provided in the Task-IL setting (making it a noticeably easier problem to solve).

<table>
<thead>
<tr>
<th>Table 1: Baselines</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline</td>
</tr>
<tr>
<td>-----------</td>
</tr>
<tr>
<td>ZERO-SHOT</td>
</tr>
<tr>
<td>JOINT</td>
</tr>
</tbody>
</table>

4.1 Experimental Results

In this section, we experiment with the use of momentum-based weight interpolation in three standard CL method categories: fine-tuning (pure SGD [37]), regularization-based (oEWC [36]), and rehearsal-based (DER++ [4] with buffer size 500 and 5000).

The results presented below are obtained over three seeds, alongside which we provide the zero-shot lower bound (Tab. 1). Interestingly, the non-adapted zero-shot performance already in parts vastly outperforms comparable adaptation with state-of-the-art methods not relying on foundation models, with e.g. DER++ [4] reporting 72.70 ± 1.36% with a buffer of 500, and 85.40 ± 0.49% with a buffer of 5000 on CIFAR-10, while zero-shot performance of our foundation model already achieves 88.77%. This difference is even further exacerbated on Tiny-ImageNet, with 19.38 ± 1.41% and 39.02 ± 0.97 for buffer sizes of 500 and 5000 respectively, versus 58.53% for zero-shot performance, verifying the potential [21, 31] of foundation models in CL.

To provide an upper bound, we train on all tasks jointly (Tab. 1). Since joint training is evaluated without task boundaries, this upper bound does not hold for Task-IL scenarios. Next, in Tab. 2 we present the results on the CL benchmarks. We empirically show that, as motivated in Sec. 3, keeping a momentum-interpolated version of the foundation model results in consistent improvements.

In particular, our results show that adaptation to the task distribution at hand is beneficial even with simple fine-tuning. Even when accounting for a change in learning rate (as noted in [4] and done for every baseline), we find that additional momentum-based weight interpolation offers consistent benefits in both class- and task-incremental settings, with nearly +4% improvement on both Seq-CIFAR-100 and Seq-Tiny-ImageNet. Furthermore, through momentum-updating, we can push simple fine-tuning close or even over the performance of a state-of-the-art CL framework (DER++).
Additionally, we observe similar performance improvements even when applied on top of separate CL frameworks, both memory-free (oEWC, e.g. 74.07 ± 0.20 → 77.25 ± 0.31 on Seq-CIFAR-100) and memory-based (DER++ with 500 memory samples, 76.78 ± 0.23 → 82.01 ± 0.31).

Interestingly, a momentum-extended DER++ with a buffer size of 500 also almost closes the gap in performance to the non-momentum based DER++ with a much larger buffer size 5000, which, even with such a large memory, also sees significant improvements on the particularly more complex CL tasks (Seq-Tiny-ImageNet, 76.54 ± 0.10 → 78.26 ± 0.14).

This demonstrates that the need for buffer sizes in CL frameworks built around foundation models can decrease significantly (in this case, 10-fold) through momentum-based weight-space interpolation. We do note that while not necessary for the benchmarks at hand, longer task sequence may benefit from a re-synchronization of $\theta_{\text{slow}}$ and $\theta_{\text{fast}}$.

Finally, we find that momentum-based DER++ with a buffer of 5000 even further closes the gap to the joint optimization upper bound - looking at the error, we find a drop of 0.45% → 0.32% on Seq-CIFAR-10, 4.06% → 2.28% on Seq-CIFAR-100, and 2.32% → 0.6% on Seq-Tiny-ImageNet, which marks a nearly 75% reduction. Conclusively, these results indicate the significant benefits of retaining a momentum-updated model copy when introducing foundation models into the CL setting, both for consistent relative improvements, but also to minimize the performance drop when moving from the standard joint optimization to a continual learning scenario.

5 Conclusion

This work tackles the adaptation of large-scale pre-trained zero-shot models to continual learning (CL). To retain the strong generalizability and robustness of these models even under continuous fine-tuning, we propose the use of a momentum-based interpolation between a slow-moving zero-shot model excluded from the direct CL process and a task-adapted fast variant. Through this simple extension, we find consistent improvements in performance across three standard CL benchmarks (Seq-CIFAR-10, Seq-CIFAR-100, Seq-Tiny-ImageNet) on both memory-based and memory-free approaches, of in parts more than $+4\%$. In addition, we find the distance between continual learning and joint task optimization performance in some cases to even be more than halved. Based on these insights, the generalizability of large-scale pre-trained zero-shot models, and the simplicity of the proposed setup, we believe the adoption of our approach to be of high practical interest.

Acknowledgements

Karsten Roth thanks the International Max Planck Research School as well as the European Laboratory for Learning and Intelligent Systems (ELLIS) PhD program for support. Zeynep Akata acknowledges partial funding by the ERC (853489 - DEXIM) and DFG (2065/1 - Project number 390727645) under Germany’s Excellence Strategy.

References


[39] Guangyuan SHI, JIA XIN CHEN, Wenlong Zhang, Li-Ming Zhan, and Xiao-Ming Wu. Overcoming catastrophic forgetting in incremental few-shot learning by finding flat minima. In M. Ranzato, A.


A Appendix

A.1 Ablation study

Momentum strength. In Figure 1 we show how the momentum strength $\tau$ affects the model’s Class-IL Accuracy. While we find that the optimal value of $\tau$ is dataset-dependent, it is encouraging that the vastly different methods show surprisingly similar behavior for a given dataset.

Restart frequency. Next, we examine whether it is beneficial to restart the fast weights $\theta_{fast}$ with the slow weights $\theta_{slow}$ during training (instead of only at the end as per default, i.e. Line 11 in Algorithm 1). To this end, we introduce a new hyperparameter restart frequency which specifies after how many gradient steps we perform a restart. From the results detailed in Figure 2, we find that restarting the fast weights is not beneficial to the generalization performance.

Update frequency. Finally, we examine whether it is beneficial to perform the update of the slow weights (Line 7 in Algorithm 1) at various frequencies. For this purpose, we introduce a new hyperparameter update frequency which specifies after how many gradient steps we update the slow weights. From the results summarized in Figure 3, we find that updating at frequencies higher than 1 (where 1 is the default behavior of our algorithm) does not provide a boost in performance.