MSc Artificial Intelligence
Master Thesis

Meeting in the middle: Embracing the stability gap with balanced losses

by
Thomas Paul Alexandre Wiggers
12114960

June 29, 2024

48EC
November 2023 - June 2024

Supervisors:
MSc. Tejaswi Kasarla
MSc. Melika Ayoughi
Examiner:
Dr. Pascal Mettes
Second reader:
MSc. Tejaswi Kasarla

Universiteit van Amsterdam

Contents

Abstract

A key challenge in incremental learning is to mitigate the forgetting of previously learned tasks when learning a new task. We focus on tackling this challenge in the exemplar-free general continual learning setting, as to obtain a method capable of generalizing to other settings. In this work we provide insight into the effects of the stability gap in incremental learning. Additionally, we introduce a method to quantify the task bias frequently observed in methods. Finally, we argue the importance of a representation learning perspective in the stability-plasticity dilemma. Building on this, we propose a novel approach, Model in the Middle, which mitigates the stability gap using a separate model, and balances stability and plasticity by using only knowledge distillation. We evaluate this approach in the exemplar-free setting and obtain state-of-the-art results in terms of final average accuracy, forgetting, average minimum accuracy and task bias. Additionally, we show its ability to generalize to relaxed settings often explored in other work, and provide insight into our methods ability to obtain general features.

Chapter 1

Introduction

Incremental learning is a behavior often observed in nature, where animals and humans alike process new information and adapt their knowledge of the world without forgetting past information and experiences. This kind of learning might benefit real world applications of AI models, such long term deployment of robotics who might need to understand and react to changes in the world such as a nearby road closure or broken home appliance. Adapting to this information without processing a full experience history and without forgetting all previous training is likely crucial.

This ability is formalized in the setting of continual learning, or lifelong learning, where the objective is to train a model on a sequential stream of data, without full access to the entire history of the data stream. This necessitates methods that maintain the knowledge of these past samples without observing them again. If this not is accounted for, *catastrophic forgetting* [16] occurs. Without any incentive not to forget, models have a strong tendency to quickly forget relevant feature representations and output distributions of past data, causing a significant drop in performance on these samples [41].

A sub-field of incremental learning is class incremental learning, which implements the incremental setting as one where a subset of the classes is presented at each training increment, known as a task, and the objective is to maximize performance over all classes. This field notes additional difficulty over other incremental learning fields as the task identity is not given at inference time, and must be inferred by the model. That is, the test image may be from any of the tasks and this task id is not given.

The term *plasticity* is used to describe the model's ability to adapt to the new samples. In order to adapt, the model must allow some part of its parameters to accommodate this change. Doing so while retaining the ability to correctly model the past samples is coined network *stability*. When performing a model update to adapt to new samples we are limited by the finite model capacity, and must choose which information to retain and which to forget and convert to new information. This *stability-plasticity dilemma* is the main challenge faced in incremental learning, and has been identified as the main driver of forgetting [20, 11, 42, 13]. Regular supervised training on all data, named joint training in incremental learning, shows us strong performance across classes is attainable with the given model capacity. While some forgetting is clearly inevitable, most continual methods fall significantly short of joint training performance, indicating significant challenges remain.

Recent work by de Lange et al. [12] investigates the effects of the incremental training mode more closely, and identify the stability gap. Disentangling the computed loss gradient into a stability and plasticity term, they show for most methods that *at the task boundary* significant imbalance in the gradient terms is observed. That is, the optimization steps following shortly after new samples are introduced suffer from a significant imbalance in these loss terms. This causes the initial updates to significantly affect previous task performance, which is subse-

quently recovered but only to a certain extent. De Lange et al. hypothesize this re-learning to be due to excessive network plasticity and contribute to excessive forgetting. This is in contrast to the recent work or [42, 20] who argue most methods rely too strongly on network stability.

Recent work [19, 44, 1] has additionally identified the imbalance of samples available during each task as a significant factor affecting the final performance. Evidently, although the dataset as a whole is balanced with respect to the classes, due to the problem formulation we can sample only classes from the current task and none of the classes in other tasks. We define this as *temporal class imbalance*, which is governed by at which time, and therefore task, we sample. Additionally, we may control this imbalance to an extent with the use of an exemplar buffer to replay past task samples. Consequently, this imbalance often leads to a bias towards the classes in the most recent task. Avoiding this *task bias* yields strong performance improvements as shown by [4, 19, 44].

Incorporating these insights, we propose a method, Model in the Middle (MITM), to separately optimize the current task and the integration of the current task with previous tasks. By performing these inter-task and intra-task optimizations separately, we can avoid imbalanced loss terms and perform balanced gradient steps to obtain features that are not biased towards any specific task. Additionally, we show this to work well in the exemplar-free variant of general continual learning (GCL) [13], and show this setting generalizes well to other proposed formulations of class incremental learning. We investigate the stability-plasticity balance needed to promote task specific features into task agnostic 'general' features, and hypothesize that these unbiased features are closer to the general features found in the model trained jointly on all data.

The contributions of this work can be summarized as:

1. Underpinning the necessity of exemplar-free general continual learning to obtain general methods.
2. Introducing a metric to quantify the task bias observed in most methods.
3. Introducing a novel method, Model in the Middle, which
    i. Balances stability and plasticity by using only distillation.
    ii. Mitigates the stability gap using a separate leading model.
4. Providing a starting point for future research on the stability-plasticity trade-off.

Chapter 2

Related work

The incremental learning field is characterized by a broad range of approaches, often borrowing significantly from other fields such as unsupervised learning, generative models or foundational models. Moreover, these methods differ strongly in their choice of problem formulation and evaluation setup. This is especially the case in class incremental learning, as it has proven more difficult than other settings [42]. In this section we place works in class incremental learning in relation to our setup and discuss the main challenges in the field.

2.1 Methods

De Lange et al. [13] categorizes the methods based on their approach focusing on parameter regularization [21, 46], parameter extension [2, 36], improving the the exemplar buffer [5, 4, 33], or any combination of these. Additionally, we can identify two main directions of research.

The most intuitive direction works on improving the approximation of the error towards the past tasks. As incremental learning does not allow us to access past samples, but requires us to maintain accurate classification of these, it is a core challenge to find some method of approximating the classification error on these samples as it is not available directly. For example, LwF[24] distills responses of a past copy of the model as the approximation of the previous samples, while DER[5] maintains a small replay buffer combined with past prediction distributions as distillation targets. This direction is promising, as obtaining a sufficient approximation would solve the problem directly, as it would be equivalent to having access to these past samples.

As noted by [22, 41], most work falls significantly short of the performance obtained when training jointly on all samples, especially without replay or generative strategies. Additionally, a strong bias towards the classes in the most recent task is observed if not corrected for [4, 19]. There are a few recent works [1, 4, 19, 44] that also directly address this bias.

Additionally, class incremental methods can be further separated in terms of of exemplar-free or exemplar-based incremental learning. The objective in incremental learning is to do so without exemplars, however due to the difficulty of this setting most work investigates the relaxed setting allowing for a small amount of exemplars [13, 41].

2.1.1 Exemplar-free methods

One of the first works in exemplar-free class incremental learning is Learning without Forgetting, specifically its single head variant **(LwF.MC)**[24, 33]. This method aims to retain a previous model's activation's on past class logits to prevent forgetting by using knowledge distillation. This provides an approximation to the past samples without requiring access to these, but

in practice is less effective than exemplar-based methods [33]. Learning without Memorizing **(LwM)**[14] extends LwF with distillation of the attention maps.

Parameter regularization methods, such as elastic weight consolidation and its extension **(oEWC)**[21, 37] and Synaptic intelligence **(SI)**[46] attempt to delegate necessary parameter change to those parameters estimated to be less important. Memory Aware Synapses **(MAS)**[3] attempts to answer the question of parameter importance based on the sensitivity of the output function to changes in the parameters on a separate dataset. MUlti-Classifier **(MUC-MAS)**[26] extends this by using an ensemble to estimate the parameter importance.

In contrast to these methods, we do not attempt to explicitly regulate individual parameters, instead we aim to provide an optimization objective that is balanced with respect to the old and new tasks.

Exemplar-free general continual learning In this work, we focus on a simple question. *If a ResNet18 can jointly train on a classification task and obtain high accuracy, why can it not do so incrementally?* In contrast to the early work discussed above, most of the recent work does not directly address this question. This is due to dependence on modification to the structure of the model [2, 36, 25, 17], a base training step with half of the dataset [50, 30, 52, 51, 25, 17], pre-training of the model [28, 47], the use of pre-trained transformers [43, 39], intermediate unsupervised modeling steps [15, 9] or generative modeling methods [38, 45].

To remain focused on this question, we propose to extend the idea of general continual learning as in [13] to the exemplar-free setting . The exemplar-free general continual learning setting generalizes the problem of class incremental learning to one with an infinite stream of non-stationary data to which a model is fitted, and at any point during this stream the model may be evaluated. This general setup does not allow for dependence on any factor such as a base step, pre-training, model extensions, exemplars or test-time task identifications. As observed in literature [13, 31], dependence on any of these often prevents results to generalize across the wide variety in evaluation setups in the class incremental learning. To avoid these dependencies, we focus on the exemplar-free general setting.

2.1.2 Exemplar-based methods

We argue the importance of exemplar-free general continual learning due to its expected ability to generalize to other settings. To show this, we compare to methods in the relaxed exemplar-based setting. For fair comparison, we only compare to exemplar-based methods that adhere to the general continual learning paradigm, excluding methods leveraging additional other dependencies such as pre-training [30] or a base step [50].

Exemplar-based incremental learning is introduced with Experience Replay **(ER)**[32, 34, 10]. The method samples from a limited buffer of past task samples in addition to the task samples, to alleviate the imbalance between current and past task samples and prevent forgetting. The method shows a strong performance improvement over incremental finetuning, and many methods extend this idea, such as [33, 4, 5, 44]. One of the few methods providing a consistent strong baseline is **(iCaRL)**[33]. iCaRL combines experience replay with knowledge distillation and uses nearest mean of exemplars to classify samples. This overcomes difficulties in incrementally constructing the classification layer due to temporal class imbalance [19].

The effect, and more importantly mitigation, of temporal class imbalance has been widely studied in methods such as **(CoPE)**[11] and others [8, 48, 4]. The iCaRL extension **(LUCIR)**[19] attribute the significant difference in embedding magnitude as the source of the resulting task bias. Similarly, in **(BiC)**[44] the authors quantify this bias in the final linear layer using a held-out validation set, and train a bias correction layer to correct for this. They show a strong performance improvement of 20 percentage point, and indicate the linear layer

has significant bias. Following this line of reasoning, we infer that the remaining 10 percentage point performance gap with respect to the upper bound can be attributed to bias in representations obtained from the feature extractor.

Another body of work focuses directly on improving the approximation of the error on samples from past tasks. Dark Experience Replay **(DER)** [5] shows us storing the predicted logits as a label along with the sample in the replay buffer, helps prevent forgetting. DER is one of the few methods which consistently outperforms iCaRL across settings, underwriting the strength of this variant of knowledge distillation. This strong baseline is extended with **(X-DER)**[4]. The authors extend the methods with modification of the stored logits, to update the logits learned after the initial computation, and avoid task bias by following [1].

Using a separate loss for the replay buffer and the incoming data in order to prevent representations of new classes overlapping older classes, and causing perturbations to the latter, is investigated in [6, 1, 29]. Notably, Asymmetric Cross-Entropy **(ER-ACE)** [6] shows very little forgetting occurs at the task boundary, effectively mitigating the stability gap. This method is similar in idea to ours, however we propose even further separation as we delegate the separate losses to separate models entirely.

When considering exemplar-based methods, we observe a dilemma. As using exemplars is a relaxation of the strict setting of incremental learning, it is an additional objective of exemplar-based methods to require as little exemplars as possible [49]. In practice, for all exemplar-based methods we observe a reduction in performance under stricter memory budgets [5, 8, 31]. While performance is generally better than exemplar-free methods [22], most exemplar-based methods cannot function without these, and therefore cannot eliminate the buffer. Instead of depending on a buffer we aim to avoid, it might be more promising to develop exemplar-free methods directly.

2.2 Challenges

Even the most advanced methods previously discussed fall significantly short of the joint training upper bound in the most strict settings. Drawing on recent insight, we can attribute this to three main challenges in addition to providing an approximation to the past tasks.

2.2.1 Temporal class imbalance

It is evident that the problem formulation of class incremental learning introduces imbalance. When sampling from the dataset to obtain a batch to perform a gradient optimization step, we only obtain samples from the classes of the current task. Given this is different from traditional class imbalance, as the dataset as a whole is balanced, we name this *temporal class imbalance*.

Proving a sufficient approximation to past tasks solves this problem in principle, as it is equivalent to jointly training on all tasks, however in practice many methods observe a significant bias towards the most recent task [1, 19, 4, 44]. In LUCIR[19] and BiC [44] it is shown addressing this bias directly leads to significant performance improvements. A method for preventing this bias in knowledge distillation methods is shown in SS-IL[1], but is strongly dependent on the task id [6]. As obtaining a sufficient approximation has proven difficult [22], we must likely address the imbalance directly and providing a mechanism to do so is therefore of significant importance.

2.2.2 Stability gap

Recently, it has been uncovered the method used to measure forgetting does not provide adequate insight into the model's ability to retain knowledge. De Lange et al. [12] show most methods exhibit large amount of forgetting directly after distributional shifts, such as when a new task is observed, and this forgetting is slowly recovered afterwards. Traditional way of measuring forgetting only after learning the new task does not capture this transient forgetting at the beginning of that task. The implications of this forgetting and recovery are not yet clear [12, 18], but we might naturally question whether forgetting and re-learning is the optimal method to remember.

For replay based methods, De Lange et al. provide the following explanation. When training incrementally, new knowledge is provided by the samples from the current task, on which some classification loss is computed to provide $\mathcal{L}_{plasticity}$. We might compute the loss with respect to the past tasks $\mathcal{L}_{stability}$. We can view the objective of incremental learning as optimizing.

$$\mathcal{L}_{joint} = \mathcal{L}_{plasticity} + \mathcal{L}_{stability}$$

However we do not have access to these past task samples. Instead, the replay buffer $\mathcal{M}$ provides as subset of past samples. The classification loss on these samples provides an approximation to the stability loss we denote as $\widetilde{\mathcal{L}}_{stability}$.

$$\widetilde{\mathcal{L}}_{joint} = \mathcal{L}_{plasticity} + \widetilde{\mathcal{L}}_{stability}$$

The stability gap is observed due to a strong imbalance in the gradient signals originating from the two data sources. Having trained on previous tasks, the model is well fitted to the samples in the replay buffer $\mathcal{M}$. However, in the initial steps after the task increment, the model is poorly fitted to these unseen task samples resulting in a large $\mathcal{L}_{plasticity}$. Optimizing towards the combination of these, the small $\widetilde{\mathcal{L}}_{stability}$ and the large $\mathcal{L}_{plasticity}$, results in greedy optimization steps towards $\mathcal{L}_{plasticity}$. Over time, error is accumulated for $\widetilde{\mathcal{L}}_{stability}$ and reduced for $\mathcal{L}_{plasticity}$. This initial forgetting is then followed with a recovery of prior knowledge as $\widetilde{\mathcal{L}}_{stability}$ is then jointly minimized with $\mathcal{L}_{plasticity}$.

The authors offer a similar argument for distillation, where $\widetilde{\mathcal{L}}_{stability}$ is provided by computing some distance metric between representations computed using a past snapshot of the model and the current representations. This past snapshot is often updated at the end of a task. When this past snapshot and the current model are equivalent or close, as is the case at the beginning of a new task, the outputs are identical and $||\nabla\widetilde{\mathcal{L}}_{stability}|| = 0$. This again leads to greedy $\mathcal{L}_{plasticity}$ steps and suffers the stability gap.

2.2.3 Towards general features

Additionally, recent work [50, 30, 52, 51, 25, 17] trends to using a base step which includes half of the classes in the dataset. This differs from the evaluation setup presented in iCaRL[33], where classes are equally divided over the number of tasks. The dependency of this base step does not allow the model to be evaluated at any time, but more importantly we argue this setup allows for too much dependence on stability. The features learned on half of the dataset likely generalize well towards the other half, significantly reducing the need to incrementally construct general features across tasks. We extend this reasoning to methods that depend on a priori pre-training.

Kim et al. [20] identify such a strong bias towards stability in most recent methods. They share our concern, but justify the base step choice with the argument that the feature extractor trained on this base step still requires improvement to rival joint training, necessitating incremental improvement of the feature extractor. We do not disagree with this conclusion,

but would like to point out both cases, with and without base step, require us to improve the feature extractor incrementally. However, we expect final performance to be significantly lower without the base step if the method is not effective in incremental improvement of the feature extractor. *This would allow for more contrast between the methods that are more effective at learning representations incrementally, and those more effective at using the prior representations.* Specifically, learning to generalize features for both inter- and intra-task discrimination is argued to be an important consideration in [42, 29]. As general continual learning does not allow for this base step, this steers us towards methods better able to incrementally train a general feature extractor.

Chapter 3

Method

In contrast to the observation in Wang et al.'s [42] survey that class incremental learning methods should focus on the facilitation of plasticity, recent work from De Lange et al. [12] argues the stability gap resulting from network plasticity is a key driver of forgetting, exemplifying the relevance of the dilemma. In this section we first present a hypothesis on the effects of the stability gap before building our method on this insight.

3.1 The importance of the stability gap

The question remains whether the stability gap itself is detrimental. In their experiments, De Lange et al. also observe decreased performance with increasing stability gap. However their experiment setup makes use of a replay buffer, therefore not isolating the effect of the stability gap from the effects of the approximation of $\widetilde{\mathcal{L}}_{joint}$ with the replay buffer.

We can investigate these effects separately in the additional baseline setting introduced by Hess et al. [18], the **Cumulative** approach[1]. In this setting, we incrementally train on the current task and all past tasks jointly, but not the future tasks. This does not adhere to the problem formulation of class incremental learning as it requires access to all of past task samples, but provides the incremental training setting *without* needing to approximate the past task samples.

Using the cumulative approach, which optimizes $\mathcal{L}_{stability}$ directly, we observe an almost complete recovery of the previous task performance. This indicates that in the presence of $\mathcal{L}_{joint}$ we do not directly observe significant detrimental effects of the stability gap on the final average accuracy. However, obtaining such an adequate approximation is a key challenge.

It appears that the stability gap is most harmful when combined with a poor approximation to the joint loss. Given the small size of the replay buffer and heavy use of oversampling, it is likely the buffer provides a poor approximation to the joint loss, providing an explanation for the significantly decreased performance observed with experience replay.

We provide a visual representation of this insight in Figure 3.1. In this schematic, we show both $\mathcal{L}_{stability}$ and its approximation $\widetilde{\mathcal{L}}_{stability}$. Any optimum for $\mathcal{L}_{stability}$ also provides a good fit for $\widetilde{\mathcal{L}}_{stability}$. However, due to the low number of samples in the replay buffer $\widetilde{\mathcal{L}}_{stability}$, many optima for $\widetilde{\mathcal{L}}_{stability}$ exist, and not all of these coincide with the optima of $\mathcal{L}_{stability}$. We visualize the effect of the stability gap on the parameter, moving from θ_{t_0} to θ_{t_1}. The recovery then moves the parameter to θ_{t_2}. It is difficult to obtain the optimum from $\mathcal{L}_{stability}$ using $\widetilde{\mathcal{L}}_{stability}$, as the recovery attempts to do, this would require us to generalize from the exemplars to the entire dataset perfectly to avoid the many optima that are sub-optimal with respect to $\mathcal{L}_{stability}$. While more difficult to quantify, we hypothesize a similar effect exists for knowledge distillation.

[1] We reproduce this experiment in Appendix Table A.3.

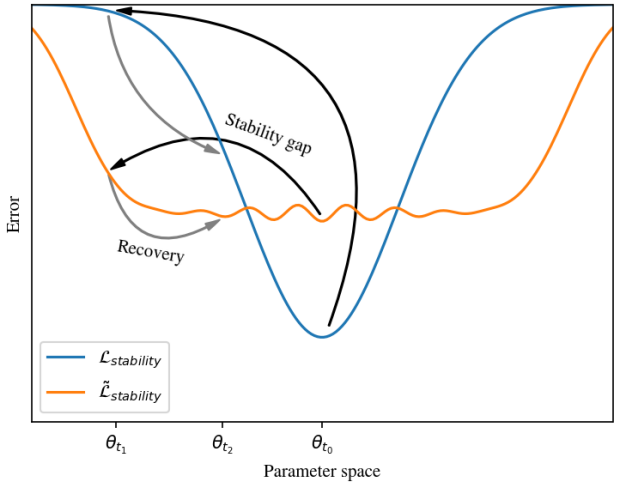


Figure 3.1: Hypothesis of joint effects of the stability gap and approximation to $\mathcal{L}_{stability}$. The stability gap moves the parameters from θ_{t_0} to θ_{t_1}, indicated with the black arrows. The optimization with respect to $\widetilde{\mathcal{L}}_{stability}$ then updates the parameters to θ_{t_2}, the gray arrows, which recovers with respect to this objective but not to the true $\mathcal{L}_{stability}$, which is observed as forgetting.

From this perspective, it appears counter-intuitive to recover the optimum of $\mathcal{L}_{stability}$ using $\widetilde{\mathcal{L}}_{stability}$. Instead, we might view the current parameters θ as prior knowledge of the optimum of $\mathcal{L}_{stability}$. If we avoid the stability gap, we significantly reduce our dependence on optimizing $\widetilde{\mathcal{L}}_{stability}$ directly to remain near the optimum of $\mathcal{L}_{stability}$.

More concretely, we might be able to prevent forgetting even if we have a poor approximation by avoiding the stability gap.

3.2 Feature bootstrapping

In this section we aim to provide an additional perspective on the the problem of class incremental learning as one of feature bootstrapping. So far, we have placed the focus on obtaining the highest accuracy across all tasks and classes, by achieving the least amount of forgetting. Most methods employ knowledge distillation or experience replay to maintain the features that have been learned, on the basis that maintaining these prevents forgetting. To a certain extent, this leads us to the assumption these features are optimal, and by extension good candidates for general features.

We hypothesize that this is assumption may not necessarily hold. The learned features are likely optimal to discriminate classes within-task. However, for optimizing these features further for inter-class discrimination entails optimizing with respect to both $\mathcal{L}_{plasticity}$ and $\widetilde{\mathcal{L}}_{stability}$. In practice we observe significant imbalance between $\mathcal{L}_{plasticity}$ and $\widetilde{\mathcal{L}}_{stability}$, both due to temporal class imbalance and the stability gap. This results in a bias of the feature representations towards the most recent task.

As argued above, if the goal is to obtain general task-agnostic features which help discriminate any class from any other, we must bootstrap these general features from the task specific ones. From this point of view, we do not attempt to maintain the task optimal learned parameters, but rather find new parameters which incorporate the discriminative power of both. These features, building on classes across tasks, progress to be more general features, in turn enabling better model performance.

3.3 Combining insights

From these perspectives, we identify a direction for exemplar-free continual learning. Additionally, we aim to comply with general continual learning, to prevent over-dependency on any specific factor. This direction consists of the following objectives:

- Avoid the stability gap to reduce the necessity of a good approximation $\widetilde{\mathcal{L}}_{stability}$
- Mitigate temporal class imbalance to prevent task bias in the representations and classifier
- Bootstrap inter-task feature representations from intra-task features.

From the literature, we can observe it is difficult to provide an adequate $\widetilde{\mathcal{L}}_{stability}$ term to balance $\mathcal{L}_{plasticity}$, in large part due to temporal class imbalance and the stability gap. This results in significant forgetting and task bias, or forces methods to depend on factors outside of the general setting such as pre-training.

We propose a new method Model in the Middle (MITM), shown in Figure 3.2, which avoids this imbalance by using only knowledge distillation. This way, upon distributional shifts, we avoid the large loss magnitudes observed when applying a cross entropy loss directly. Additionally, *using the same mechanism for stability and plasticity may simplify the search for a balance between the two.* We therefore reformulate the objective as

$$\widetilde{\mathcal{L}}_{joint} = \widetilde{\mathcal{L}}_{plasticity} + \widetilde{\mathcal{L}}_{stability}$$

To incorporate knowledge of the current task, we train a separate *leading* network which optimizes only the current task using cross entropy. The leading model exhibits catastrophic forgetting with respect to the previous tasks. This model is used to provide $\widetilde{\mathcal{L}}_{plasticity}$ through knowledge distillation. The middle model distills this new knowledge from the leading model by matching its logits of the new classes using mean squared error, in similar fashion to [5]. At the task boundary, the middle model's parameters are copied to trailing model to retain the new knowledge. This dependency on the task boundary does not comply with general continual setting and is addressed later. Distillation from this trailing model ensures the stability on previous classes, again using mean squared error on the previous classes' logits.

This setup offers nice properties with respect to the magnitudes of the error terms with respect to distributional shifts, e.g. task increments. When new classes are observed, the leading model exhibits significant prediction error. However, this is only corrected slowly through many gradients steps. Shortly after the task increment, middle model's output still resembles the leading model's, as this has been optimized throughout the previous task. As the leading model shifts towards adapting to the new data distribution, the middle model can slowly follow. In this position, the loss magnitude of both is balanced, allowing the model to optimize towards features that are optimal for both trailing and leading classes. The intuition is this allows general features to be constructed in a balanced way from the class-specific features of the leading model and the previous features in the trailing model.

3.4 Model in the Middle

The incremental learning problem is split up into $N = |\mathcal{T}|$ tasks denoted with the set $\mathcal{T} = \{T_1, ..., T_c, ..., T_N\}$. We use $T_c \in \mathcal{T}$ to indicate the current task. The dataset $\mathcal{D}$ is split into an equivalent number of subsets $\mathcal{D}_{T_c} \subset \mathcal{D}$ and a held-out evaluation set $\mathcal{E}_{T_c} \subset \mathcal{E}$. A task consists of all samples and labels $(x, y) \sim \mathcal{D}_{T_c}$ in a task. Classes in a task $\mathcal{C}_{T_c} \subset \mathcal{C}$ are mutually exclusive with all other tasks s.t. $\mathcal{C}_{T_i} \cap \mathcal{C}_{T_j} = \emptyset$. The evaluation accuracy for this task is computed on

$(x, y) \sim \mathcal{E}_{T_c}$. The samples drawn from these tasks conform to the i.i.d. assumption. This is in contrast to the dataset $\mathcal{D}$ as a whole, as it presents the tasks sequentially over time, thereby undermining the stationarity assumption.

The standard objective is to find parameters θ of the function $f(\cdot;\theta)$ to minimize

$$\min_{\theta} \mathcal{L} = \mathbb{E}_{(x,y)\sim\mathcal{D}} \left[\ell\left(f(x;\theta), y\right)\right]$$

where $\ell(\cdot,\cdot)$ denotes the soft-max cross-entropy loss function. This is problematic when approximating the expectation, as $\mathcal{D}$ is not a stationary distribution. We therefore sequentially minimize the stationary tasks

$$\min_{\theta} \sum_{i=1}^{N} \mathcal{L}_{T_i}$$

and introduce

$$\mathcal{L}_{T_i} = \mathcal{L}_{trailing} + \lambda \mathcal{L}_{leading} + \mathcal{L}_{classification} \tag{3.1}$$

The parameter λ controls the balance between the leading and trailing distillation loss, and provides control over the plasticity-stability trade-off.

Our approach uses two models in addition to θ. We denote the parameters of the leading model with ϕ and the trailing model as ψ. We optimize the parameters of the leading model directly and apply standard soft-max cross-entropy function ℓ. We denote with y the one-hot encoded label, predicted logits $a = f(x;\theta)$, and $|\mathcal{C}_{T_c}|$ the number of classes in the current task T_c.

$$\ell(a, y) = -\sum_{i=1}^{|\mathcal{C}_{T_c}|} y_i \log\left(\frac{e^{a_i}}{\sum_{k=1}^{N} e^{a_k}}\right) \tag{3.2}$$

The classification loss term 3.3 applies to the leading model. The loss is applied only to the logits of the classes in the current task $\mathcal{C}_{T_c}$. We denote the logits of the classes in task T_c with $f_{T_c}(x;\phi)$ and the logits of all past tasks up to T_c with $f_{T_{0;c-1}}(x;\phi)$.

$$\mathcal{L}_{classification} = \mathbb{E}_{(x,y)\sim\mathcal{D}_{T_c}} \left[\ell\left(f_{T_c}(x;\phi), y\right)\right] \tag{3.3}$$

For the plasticity term 3.4 we apply knowledge distillation by matching the logits of the current classes from the middle model θ to those from the leading model ϕ using the mean squared error, similar to [5]. Note, the leading model is only optimized through its respective loss in eq 3.3, as we do not propagate the gradients from this loss to ϕ.

$$\mathcal{L}_{leading} = \mathbb{E}_{(x,y)\sim D_{T_c}} \left[\frac{1}{n}\sum_{i=1}^{n}(f_{T_c}(x;\theta) - f_{T_c}(x;\phi))^2\right] \tag{3.4}$$

For the stability term 3.5 of the trailing model we apply the same logit matching, but only on the logits from the previous task classes. As with the leading distillation loss, we do not propagate gradients through the trailing model. The parameters of the trailing model are therefore only modified upon model copy, although we allow the trailing model to update its batchnorm statistics [40]. Note for the first task, this term does not contribute as there are no past logits.

$$\mathcal{L}_{trailing} = \mathbb{E}_{(x,y)\sim D_{T_c}} \left[\frac{1}{n}\sum_{i=1}^{n}(f_{T_{0;c-1}}(x;\theta) - f_{T_{0;c-1}}(x;\psi))^2\right] \tag{3.5}$$

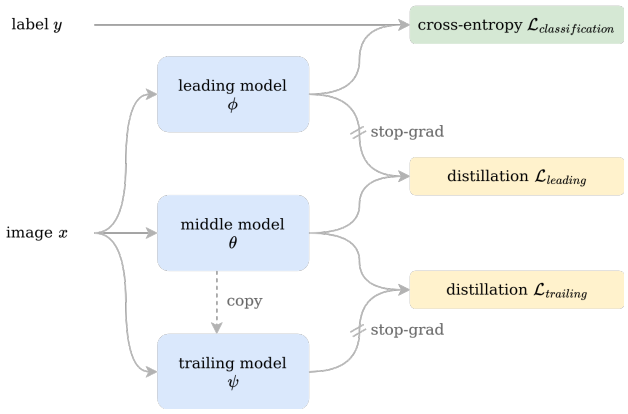


Figure 3.2: Overview of Model in the Middle. We balance stability and plasticity by using only knowledge distillation, and incorporate new knowledge through a separate leading model. This model exhibits harmless catastrophic forgetting and buffers sudden distributional changes to mitigate the stability gap. No gradients are propagated through connections labeled stop-grad.

3.4.1 Using exemplars

In section 2.1.1 we outline the necessity of focusing on exemplar free general continual learning. However, to demonstrate the ability to generalize to the exemplar-based setting, we investigate the addition of exemplars. To add exemplars, we modify the expectation of equations 3.4, 3.5 and 3.3 to be

$$\mathbb{E}_{(x,y)\sim D_{T_c}\cup\mathcal{M}} \tag{3.6}$$

As in [10, 5, 44] and others, we approximation the expectation by sampling half the batch from D_{T_c} and the other half $\mathcal{M}$, as to avoid class imbalance from the replay buffer. Additionally, we modify eq 3.1 to add a cross entropy loss 3.7 on the labels for the middle model directly. To avoid introducing the stability gap, we scale this by the fraction of epochs that have passed for the current task $\eta \in [0, 1]$. This way, when large distributional changes are occur at the task boundary, η is (close to) zero.

$$\mathcal{L}_{middle} = \eta \cdot \mathbb{E}_{(x,y)\sim D_{T_c}\cup\mathcal{M}} \left[\ell(f_{T_c}(x;\theta), y)\right] \tag{3.7}$$

3.4.2 General continual learning

At the task boundary, the middle model's parameters are copied to trailing model to retain the newly incorporated knowledge. To remove the dependency on the task boundary, this can also be performed using an exponential moving average. After every epoch we perform the update

$$\psi = \psi \cdot (1 - \alpha) + \theta \cdot \alpha \tag{3.8}$$

This is not overly sensitive to the choice of $\alpha \in (0.9, 1)$. Our method remains dependent on the task boundary to determine whether the trailing or leading model should provide the target for a given logit. This dependency can be removed by keeping a record of which classes are currently in samples available for training and assigning these logits to the leading model, and past logits to the trailing model. While this solution to an extent simply recovers the task boundary, we expect a more real-world application to require a custom solution for this aspect in any case.

As our evaluation datasets allow us to easily recover the task boundary, and due to time constraints, we continue with our task boundary dependent version. A short evaluation of the general version is provided in the ablations in Table 4.4b.

3.5 Evaluation

We follow the evaluation metrics as in [12]. For every task T_k we define a training dataset $\mathcal{D}_{T_k}$ and a held-out evaluation set $\mathcal{E}_{T_k}$. We define the accuracy $\mathbf{A}(\mathcal{E}_{T_k}, f(\theta_{T_k})) \in [0, 1]$ as the percentage of correct top-1 classifications on evaluation set $\mathcal{E}_{T_k}$ with the parameters θ_{T_k} just after training on task T_k.

Average Accuracy (ACC) is the accuracy, after learning task T_k, over all evaluation tasks $\mathcal{E}_{<k} \cup \mathcal{E}_k$ of tasks seen so far. We define average accuracy as in [12]:

$$\mathbf{ACC}_{T_k} = \frac{1}{k} \sum_{i=1}^{k} \mathbf{A}(\mathcal{E}_{T_i}, f(\theta_{T_k})) \tag{3.9}$$

In this work, we mainly report the Final Average Accuracy (**FAA**), equivalent to the average accuracy $\mathbf{ACC}_{T_N}$ across all evaluation tasks after the last training task. As shown by [12, 10, 6] and others, this overlooks the performance of the model *during* training, which may significantly vary. We provide our results using FAA and give insight into the accuracy during training using the following metrics:

Forgetting is the forgetting $\mathbf{F}_{T_j}^{T_i}$ on task j after the model is trained on task i. It is defined as in [10]:

$$\mathbf{F}_{T_j}^{T_i} = \max_{l \in \{1, \cdots, i-1\}} \mathbf{A}(\mathcal{E}_{T_l}, f(\theta_{T_j})) - \mathbf{A}(\mathcal{E}_{T_i}, f(\theta_{T_j})) \tag{3.10}$$

The average forgetting measure at task T_k is then defined as:

$$\mathbf{F}_{T_k} = \frac{1}{k-1} \sum_{j=1}^{k-1} \mathbf{F}_{T_j}^{T_k} \tag{3.11}$$

Average minimum accuracy We define min-ACC as in [12]. This metric quantifies the stability gap, by giving a measure of the lowest absolute accuracy observed for past tasks during the training of later tasks. It provides an additional insight into the worst case accuracy of the model *during* training, which might be missed by forgetting and final average accuracy. It is defined as the average lowest accuracy for all previous tasks:

$$\mathbf{min\text{-}ACC}_{T_k} = \frac{1}{k-1} \sum_{i=1}^{k-1} \min_{n} \mathbf{A}(\mathcal{E}_{T_i}, f(\theta_n)), \ \forall t_{|T_i|} < n \leq t_{|T_k|} \tag{3.12}$$

where the iteration number n ranges from the iteration after the task T_i is learned at $t_{|T_i|}$ until current iteration $t_{|T_k|}$. With θ_n we indicate the model parameters at training iteration n during training.

Task accuracy deviation Additionally, we report the standard deviation of the per-task final accuracy. With this metric we aim to provide insight into the task bias of the methods. The intuition is that a method with significant bias towards any task has a larger variance in task accuracy's than an unbiased method with equal accuracy for all tasks. As it is not independent of FAA, it should always be considered in comparison to it. It is defined as:

$$\mathbf{TA}\sigma = \sqrt{\frac{1}{N_\mathcal{T}} \sum_{i=1}^{N_\mathcal{T}} (\mathbf{A}(\mathcal{E}_{T_i}, f(\theta)) - \mu)^2} \ \text{ where } \ \mu = \frac{1}{N_\mathcal{T}} \sum_{i=1}^{N_\mathcal{T}} \mathbf{A}(\mathcal{E}_{T_i}, f(\theta)) \tag{3.13}$$

Chapter 4

Experiments

Incremental learning is known to have many problem formulation variations and evaluation configurations, such as the number of epochs, memory size and pre-training. We follow a the evaluation protocol outlined in iCaRL[33], also followed by [19, 5, 4, 44].

We compare our method to other exemplar-free general continual learning methods. To show the ability of our method to generalize to other incremental settings, we also compare to exemplar-based methods. Additionally, we report results with a base step in Appendix A.1 to facilitate comparison to this relaxed setting.

4.1 Implementation details

We implement our experiments using the continual learning library Avalanche [7]. For the experiments we use the provided slimmed ResNet18 based on [27]. We modify the max pooling layer to be adaptive in size, to accommodate varying image input sizes.

For all experiments we maintain the same parameters. We follow the training and evaluation setup as in iCaRL. To comply with exemplar free general continual learning we do not use a base step, and divide all classes equally over all tasks. We use SGD with a batchsize of 32 and a learning rate of 0.01 for the leading and middle model. We train for 100 epochs per task, and do not make use of any learning rate schedule or weight regularization.

4.2 Datasets

We evaluate on three datasets widely used in class incremental learning, CIFAR10, CIFAR100 [23], TinyImageNet [35]. The CIFAR datasets contain 60000 32x32 color images in 10 or 100 classes respectively. Additionally, there are 10000 test images. TinyImageNet contains 100000 64×64 color images and 200 classes. Each class has 500 training images and 50 test images.

For all datasets we normalize the images and augment with random horizontal flips. We train all datasets for 10 tasks and all classes divided equally over the tasks, with the exception of CIFAR10 for which we use 5 tasks. We denote this split variant of the dataset as S-CIFAR10, et cetera.

4.3 Results

We compare our method to exemplar-free and exemplar-based general continual learning state-of-the-art methods. Where possible, we report the author's original result. Reproductions for DER and variants in this table have been produced with the code provided by the authors,

$\mathcal{M} = 0$	**S-CIFAR10**	**S-CIFAR100**	**S-TinyImageNet**
Joint	93.07	73.31	55.09
oEWC [37]	19.49±0.1†	-	7.58±0.1†
SI [46]	19.48±0.2†	-	6.58±0.3†
LwF.MC [33]	<u>19.61±</u>0.1†	16.22‡	8.46±0.2†
LwM [14]	-	25§	-
MUC-MAS [26]	-	<u>27</u>§	<u>22.5</u>§
MITM	**48.34±4.9**	**45.73±1.3**	**33.34±0.2**
Synthetic $\mathcal{M}$			
ABD [38]	-	43.9±0.9	-

Table 4.1: Comparison of methods with final average accuracy (FAA) as reported by the respective authors. MITM significantly outperforms other methods, even methods using synthetic data sources. Mean of 5 runs with std-dev is shown for MITM. CIFAR10: 5 tasks. CIFAR100 & TinyImageNet: 10 tasks. Additional sources and notes: †: DER, ‡: X-DER, *: reproduced with DER repository, §: estimated from a plot in the respective work. Best result in bold, second best underlined.

and best hyperparameters have been reused from the most similar dataset when not available directly. We also provide joint training as upper bound.

4.3.1 Exemplar-free

In Table 4.1 we compare MITM to other exemplar-free general continual learning methods. We report the final average accuracy across all classes at the end of training of the final task. We observe a significant improvement of MITM over other exemplar-free methods. Even in comparison to synthetic data generating methods MITM provides strong performance.

We attribute this performance to MITM's ability to balance stability and plasticity, which is simplified due to the use of only distillation losses. We visualize the stability and plasticity losses in Figure 4.1b. Both losses are in a similar order of magnitude, and crucially, strongly correlate around the task boundary. This prevents imbalance between the loss terms and the resulting stability gap and bias towards either term. We provide further support and insight into this result in subsection 4.3.3.

4.3.2 Exemplar-based

In Table 4.2 we compare MITM to other exemplar-based methods, to show the generalisability of the method to this setting. For most methods, we can observe a strong variance in performance depending on the task and number of exemplars. A generally strong method is DER, but this method does struggle in the low buffer regime, for example on CIFAR100 with $\mathcal{M} = 200$. With only two exemplars per class, performance is drastically reduced. A similar trend is observed for other methods and datasets.

MITM is a strong contender across all datasets. On CIFAR10, we are outperformed by DER, but as dataset complexity increases and number of samples per class is reduced with CIFAR100 and TinyImageNet, we significantly outperform DER and others. This is especially apparent on TinyImageNet, where in the low buffer regime we outperform others significantly, but when exemplars are added we obtain comparable results. We observe MITM is less effective at using the provided exemplars than these methods, but compensates this by leveraging its exemplar-free structure. Additionaly, in TinyImageNet we see a performance regression from

	S-CIFAR10			S-CIFAR100			S-TinyImageNet		
$\mathcal{M}$	200	500	2000	200	500	2000	200	500	5120
GDUMB[31]	35.0±0.6	45.8±0.9	-	-	9.98‡	-	-	-	-
iCaRL[33]	49.02±3.2†	47.55±4.0†	-	-	46.52‡	49§	7.53±0.8†	9.38±1.5†	7.53±0.8†
LUCIR[19]	-	-	51§	-	40.59‡	41.73‡	-	-	-
BIC[44]	-	-	-	-	36.02‡	46.39‡	-	-	-
CoPE[11]	36§	42§	51§	-	-	-	-	-	-
ER-ACE[6]	-	-	-	-	38.75‡	49.72‡	-	-	-
DER[5]	61.93±1.8	70.51±1.7	79.91*	22.02*	36.60‡	51.89‡	11.87±0.8	17.75±1.1	36.73±0.6
DER++[5]	**64.88**±1.8	**72.70**±1.4	**80.56***	25.61*	38.25‡	53.63‡	10.96±1.2	19.38±1.4	**39.02**±1.0
X-DER[4]	60.0*	67.52*	69.04*	35.66*	**49.93**	**59.14**	-	-	-
MITM	59.75±3.5	62.13±1.5	67.3±4.1	**46.60**±1.1	49.16±0.8	54.46±0.7	**32.42**±0.4	**33.06**±0.2	35.38±0.2

Table 4.2: Comparison of methods with final average accuracy as reported by the respective authors. Mean of 4 runs with std-dev is shown for MITM. CIFAR10: 5 experiences. CIFAR100 & TinyImageNet: 10 experiences. Additional sources and notes: †: DER, ‡: X-DER, *: reproduced with DER repo, §: estimated from a plot in the respective work. A dash indicates results are not available.

exemplar-free to exemplar-based, and marginal improvements when increasing the number of exemplars, further demonstrating the exemplars are not used as effectively as other methods.

4.3.3 Looking deeper

De Lange et al. [12] has highlighted the importance of evaluation metrics beyond final average accuracy. Additionally, the significant task bias present in most methods also deserves attention, as in [44, 19]. We provide deeper insight into our results with additional metrics. Along with final average accuracy, we report forgetting. We aim to minimize forgetting, but not at the cost of final accuracy, as a simple solution to prevent forgetting is not to learn at all. Additionally, Min-ACC provides insight into the amount of forgetting and re-learning taking place, as this is hypothesized to deteriorate final performance. Finally, we aim to overcome temporal class imbalance, such that we mitigate the task bias often observed in methods. To this end, we report the final task accuracy standard deviation, to provide insight into the bias of the final accuracies.

In Table 4.3 we compare our method to others using these metrics. Avalanche is used to reproduce DER, BIC and EWC, providing insight into the min-ACC of these methods. These reproductions under-perform with respect to the values given in X-DER, which are also provided for fair comparison.

We can observe MITM obtains a high accuracy in all settings. In terms of forgetting, we outperform other methods, or are close to it. These two metrics together indicate we do not prevent forgetting simply by not learning. We attribute this ability to the obtained average minimum accuracy. Compared to both DER and BiC, MITM scores exceptionally strong min-ACC, indicating successful mitigation of the stability gap. This is also visualized in Figure 4.1a, where we show the per iteration accuracy in the steps following the task increment. We can observe significantly more forgetting an re-learning of the previous task for BiC and DER, observed as the rapid decrease and recovery in accuracy, which is avoided in MITM. Avoiding this likely prevents forgetting [12, 18]. Finally, we observe a lower task accuracy standard deviation than DER. We obtain task bias closer to X-DER and BiC, methods specifically tailored to avoid task bias. This indicates our method successfully mitigates task bias in our exemplar-free configuration, with the exemplars introducing a small amount of task bias. X-DER also scores exceptionally well. We are unable to report min-ACC for this method, however

as it is similar in design to DER and takes no preventative measures against the stability gap we expect this method to also score a min-ACC comparable to DER.

For the exemplar-free case we were unable to reproduce the methods reported in Table 4.1 other than EWC, which we significantly outperform. However, when comparing exemplar-free MITM to exemplar-based methods, we see the method is very competitive, especially in terms of forgetting and task bias. We observe however that adding exemplars to MITM increases final average accuracy, but also forgetting and task bias. We hypothesize the use of exemplars, while providing an increase in final average accuracy, also introduces challenges in these regards that are not easily overcome. This could be an additional explanation for why most methods fail to generalize good performance across all settings.

In summary, the combination of these results shows us MITM does not maximize one metric at the cost of another. This is significant, as this likely allows the method to generalize well to other datasets and settings.

		FAA↑	Forg↓	min-ACC↑	TAσ↓	FAA↑	Forg↓	min-ACC↑	TAσ↓
$\mathcal{M}=500$	ER-ACE[6]	38.75	40.04	-	-	49.72	25.71	-	-
	BIC[44]	36.02	51.85	-	-	46.39	40.49	-	-
	BIC¶	35.22±1.8	**11.96**±1.9	8.45±3.9	**6.81**	42.81±1.3	**11.53**±0.9	14.87±2.4	**3.50**
	DER[5]	36.60	54.99	-	16.12‡	51.89	34.54	-	9.19‡
	DER¶	23.05±1.0	69.07±0.6	1.48±1.7	21.45	43.33±1.3	39.10±5.0	3.59±5.5	13.72
	DER++[5]	38.25	50.54	-	15.10‡	53.63	33.66	-	13.91‡
	DER++¶	24.19±1.1	68.29±1.2	0.81±0.8	22.32	44.11±0.5	42.79±0.7	5.47±6.1	13.65
	X-DER[4]	**49.93**	19.90	-	12.39‡	**59.14**	12.58	-	4.85‡
	MITM	47.31	15.23	**41.90**±12.6	10.87	56.33	20.39	**44.36**±7.3	5.88

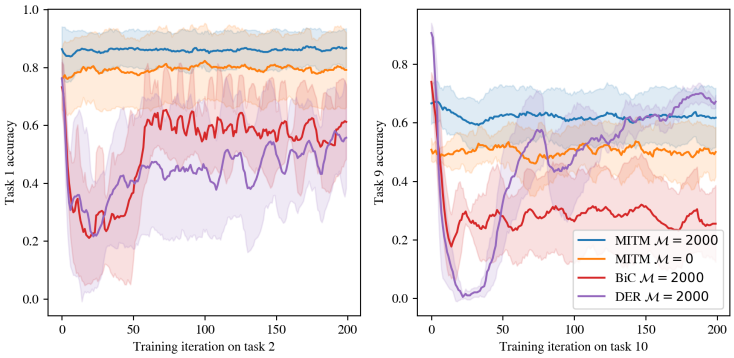
		FAA↑	Forg↓	min-ACC↑	TAσ↓
$\mathcal{M}=0$	EWC[21]¶	8.82±0.4	86.24±0.5	0.00±0.0	26.47
	MITM	**45.73**	**15.02**	**34.49**±7.0	**3.94**

Table 4.3: Final average accuracy, forgetting (FAA), average minimum accuracy (min-ACC) and task accuracy standard deviation (TAσ) on S-CIFAR100, with non-reproduced reported values by X-DER[4]. MITM scores well in all metrics, especially min-ACC. Mean of three runs. ¶: Reproduced using Avalanche[7]. ‡: Reproduced using DER repository.

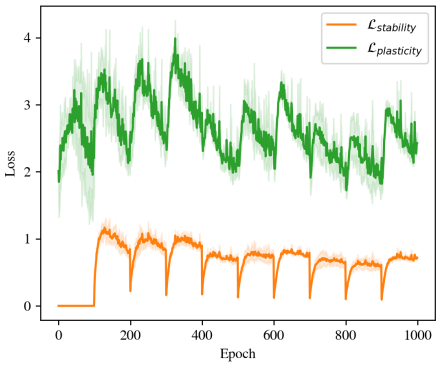
4.3.4 Feature generality

To investigate the hypothesis that we obtain more general features, we perform an experiment to measure the ability of the features to transfer to another dataset. If more general features are learned, the feature extractor should transfer better to a new dataset. We measure the transfer from CIFAR100 to CIFAR10. While similar in domain, the classes do not overlap and can still provide insight into the ability of the features to generalize to unseen classes. To decouple the ability to learn better features from their quality, we freeze the parameters of the feature extractor, and only train a new linear layer. This method is also employed by [20] to demonstrate the focus on stability by most methods.

We train the linear layer with SGD with a learning rate of 0.01 and evaluate on the test set. Results are shown in Table 4.4a. We observe features learned by MITM transfer better than other methods. This supports our argument of balanced losses preventing bias in the feature representations, especially as we outperform bias correction method BiC.



(a) Per training iteration validation accuracy as in [12] at the task boundary for first and last tasks. We observe a significant *stability gap* for methods other than MITM, where previous task accuracy significantly drops and recovers upon learning a new task. Mean and std-dev of four runs is shown.



(b) Comparison of distillation loss magnitude for MITM with and without buffer. We observe a strong correlation between the losses at the task boundary, avoiding significant imbalance.

Figure 4.1

	$\mathcal{M}$	CIFAR10 (unseen)↑
BIC[44]	2000	61.40
DER[5]	2000	64.57
MITM	2000	<u>73.01</u>
MITM	0	**74.44**

(a)

Model	FAA↑
MITM w\o $\mathcal{L}_{middle}$	47.84±0.3
Cross entropy distillation loss	37.44±0.2
Uniform sampling of $\mathcal{M} \cup \mathcal{D}_{T_c}$	42.27±2.0
efGCL MITM §3.4.2	55.32±0.5
MITM	54.46±0.7

(b)

Table 4.4: (a) Test set accuracy on CIFAR10 with frozen feature extractor after incremental training on CIFAR100 for 100 epochs. Reproduced using Avalanche. Features learned by MITM transfer better than other methods. CIFAR100 accuracy as reported in Table 4.3.
(b) MITM ablations on S-CIFAR100 with $\mathcal{M} = 2000$.

4.3.5 Ablations

In Table 4.4b we provide ablations to justify method design choices for model in the middle. We empirically found a significant reduction in the following cases: When not including $\mathcal{L}_{middle}$ for exemplar-based MITM; The use of cross entropy loss with softmax and temperature over means squared error as a knowledge distillation criterion; Not oversampling the replay buffer $\mathcal{M}$ in comparison to the task $\mathcal{D}_{T_c}$.

Additionally, we observe a slight advantage for the exemplar free general continual learning (efGCL) variant not dependent on task boundaries described in subsection 3.4.2. Due to time constraints, this version is not used for all experiments.

Chapter 5

Conclusion

Many recent works have focused on the problem of incremental learning. Few works however have focused on the *general continual learning* setting. As these recent methods are often not applicable to this setting, we argue the importance of tackling this general setting directly. Additionally, by constructing our method to be exemplar free from the ground up, we avoid the common issues observed in exemplar based methods as the buffer size is reduced.

In this work, the effects of the stability gap, feature generalisability, and temporal class imbalance are discussed. We hypothesize avoiding the stability gap reduces detrimental effects of an inaccurate approximation of the past tasks, as obtaining this approximation is a core challenge of incremental learning. A new perspective on the stability-plasticity dilemma is given through representation learning, as we formulate the goal as obtaining general task-agnostic features. Finally, under the assumption of an imperfect task approximation, we argue the need to explicitly mitigate temporal class imbalance, and introduce a metric to quantify the resulting task bias.

We propose a novel approach, Model in the Middle, which mitigates the stability gap using a separate leading model. Combining this leading model and a trailing copy, we apply only distillation losses to distill the knowledge of both in the middle model. Using this setup, we tackle temporal class imbalance with balanced stability and plasticity distillation losses, and embrace the stability gap by allowing the leading model to slowly absorb large changes in the data distribution. We show Model in the Middle to be highly beneficial in both the exemplar and exemplar-free case in terms of final average accuracy, forgetting, average minimum accuracy and task bias. The combination of these results shows us our approach does not maximize one metric at the cost of another. This is significant, as it likely allows the method to generalize well to other settings, a common hurdle for previous works. Finally, we show the learned representations transfer better than those learned by other methods, indicating our method obtains more general feature representations.

5.1 Limitations and future work

When using larger exemplar buffers Model in the Middle is outperformed by methods that make more efficient use of these exemplars. Improving the effectiveness of the exemplars in our method is likely to improve results, and to do so we might borrow insight from other methods such as X-DER[4]. Additionally, we observe that adding exemplars to Model in the Middle increases final average accuracy, but at the cost of increased forgetting and task bias. This observation warrants further investigation, as this might have implications for other methods.

Another potential limitation originates in the leading model. At the beginning of each task, the model is pre-trained on the previous task. In the cumulative setting, we can observe this pre-training to be slightly detrimental. Avoiding this might yield better features in the

leading model, and subsequently in the middle model. This idea is explored in Appendix A.4. Additionally, self-supervised learning might be leveraged to extract stronger features in the leading model. This might especially be necessary in cases where the number of classes in each increment is small. Learning to separate a small amount of classes likely requires less complex features, which in turn increases the burden on the middle model to obtain more complex inter-task features from these simpler intra-task features. Moreover, in [1] it is shown knowledge distillation instills bias into the model. Their task-wise variant might be leveraged to reduce task bias further.

Finally, *our method provides a simplification of the search for balance between stability and plasticity*, as we use the same mechanism for both in our approach. We consider the distillation losses to be balanced in part due to the excellent performance of Model in the Middle and due to this symmetry. It is possible further insight into this balance might lead to better results, and provides and exciting direction for further research into the stability-plasticity dilemma.

Acknowledgments

I would like to thank in no particular order: My supervisors, Melika and Tejaswi, for the time and guidance they have given me, Pieter Maarse for endless discussions about my thesis, Douwe van der Wal for endless coffee and idea's, Thomas van Orden and Roman Oort for endless tosti's and working sessions at CREA. This work would not have been finished without you.

Bibliography

[1] Hongjoon Ahn et al. "Ss-il: Separated softmax for incremental learning". In: *Proceedings of the IEEE/CVF International conference on computer vision*. 2021, pp. 844–853.

[2] Rahaf Aljundi, Punarjay Chakravarty, and Tinne Tuytelaars. "Expert gate: Lifelong learning with a network of experts". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 3366–3375.

[3] Rahaf Aljundi et al. "Memory Aware Synapses: Learning what (not) to forget". In: *ArXiv* abs/1711.09601 (2017). URL: https://api.semanticscholar.org/CorpusID:4254748.

[4] Matteo Boschini et al. "Class-incremental continual learning into the extended der-verse". In: *IEEE transactions on pattern analysis and machine intelligence* 45.5 (2022), pp. 5497–5512.

[5] Pietro Buzzega et al. "Dark experience for general continual learning: a strong, simple baseline". In: *Advances in neural information processing systems* 33 (2020), pp. 15920–15930.

[6] Lucas Caccia et al. "New Insights on Reducing Abrupt Representation Change in Online Continual Learning". In: *International Conference on Learning Representations*. 2022. URL: https://openreview.net/forum?id=N8MaByOzUfb.

[7] Antonio Carta et al. "Avalanche: A PyTorch Library for Deep Continual Learning". In: *Journal of Machine Learning Research* 24.363 (2023), pp. 1–6. URL: http://jmlr.org/papers/v24/23-0130.html.

[8] Francisco M Castro et al. "End-to-end incremental learning". In: *Proceedings of the European conference on computer vision (ECCV)*. 2018, pp. 233–248.

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

[10] Arslan Chaudhry et al. "Continual learning with tiny episodic memories". In: *Workshop on Multi-Task and Lifelong Reinforcement Learning*. 2019.

[11] Matthias De Lange and Tinne Tuytelaars. "Continual prototype evolution: Learning online from non-stationary data streams". In: *Proceedings of the IEEE/CVF international conference on computer vision*. 2021, pp. 8250–8259.

[12] Matthias De Lange, Gido van de Ven, and Tinne Tuytelaars. "Continual evaluation for lifelong learning: Identifying the stability gap". In: *Proceedings of The Eleventh International Conference on Learning Representations-ICLR 2023*. OpenReview. net. 2023.

[13] Matthias De Lange et al. "A continual learning survey: Defying forgetting in classification tasks". In: *IEEE transactions on pattern analysis and machine intelligence* 44.7 (2021), pp. 3366–3385.

[14] Prithviraj Dhar et al. "Learning without memorizing". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2019, pp. 5138–5146.

[15] Sebastian Dziadzio et al. "Disentangled Continual Learning: Separating Memory Edits from Model Updates". In: *arXiv preprint arXiv:2312.16731* (2023).

[16] Robert M French. "Catastrophic forgetting in connectionist networks". In: *Trends in cognitive sciences* 3.4 (1999), pp. 128–135.

[17] Xinyuan Gao et al. "CEAT: Continual Expansion and Absorption Transformer for Non-Exemplar Class-Incremental Learnin". In: *arXiv preprint arXiv:2403.06670* (2024).

[18] Timm Hess, Tinne Tuytelaars, and Gido M van de Ven. "Two complementary perspectives to continual learning: Ask not only what to optimize, but also how". In: *arXiv preprint arXiv:2311.04898* (2023).

[19] Saihui Hou et al. "Learning a Unified Classifier Incrementally via Rebalancing". In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2019.

[20] 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*. 2023, pp. 20196–20204.

[21] James Kirkpatrick et al. "Overcoming catastrophic forgetting in neural networks". In: *Proceedings of the national academy of sciences* 114.13 (2017), pp. 3521–3526.

[22] Jeremias Knoblauch, Hisham Husain, and Tom Diethe. "Optimal continual learning has perfect memory and is np-hard". In: *International Conference on Machine Learning*. PMLR. 2020, pp. 5327–5337.

[23] Alex Krizhevsky, Geoffrey Hinton, et al. "Learning multiple layers of features from tiny images". In: (2009).

[24] Zhizhong Li and Derek Hoiem. "Learning without forgetting". In: *IEEE transactions on pattern analysis and machine intelligence* 40.12 (2017), pp. 2935–2947.

[25] Yaoyao Liu, Bernt Schiele, and Qianru Sun. "Adaptive aggregation networks for class-incremental learning". In: *Proceedings of the IEEE/CVF conference on Computer Vision and Pattern Recognition*. 2021, pp. 2544–2553.

[26] Yu Liu et al. "More Classifiers, Less Forgetting: A Generic Multi-classifier Paradigm for Incremental Learning". In: *European Conference on Computer Vision*. 2020. URL: https://api.semanticscholar.org/CorpusID:226841742.

[27] David Lopez-Paz and Marc'Aurelio Ranzato. "Gradient episodic memory for continual learning". In: *Advances in neural information processing systems* 30 (2017).

[28] Mark D McDonnell et al. "Ranpac: Random projections and pre-trained models for continual learning". In: *Advances in Neural Information Processing Systems* 36 (2024).

[29] Sudhanshu Mittal, Silvio Galesso, and Thomas Brox. "Essentials for class incremental learning". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2021, pp. 3513–3522.

[30] Grégoire Petit et al. "Fetril: Feature translation for exemplar-free class-incremental learning". In: *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*. 2023, pp. 3911–3920.

[31] Ameya Prabhu, Philip HS Torr, and Puneet K Dokania. "Gdumb: A simple approach that questions our progress in continual learning". In: *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16*. Springer. 2020, pp. 524–540.

[32] Roger Ratcliff. "Connectionist models of recognition memory: constraints imposed by learning and forgetting functions." In: *Psychological review* 97.2 (1990), p. 285.

[33] Sylvestre-Alvise Rebuffi et al. “icarl: Incremental classifier and representation learning”. In: *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*. 2017, pp. 2001–2010.

[34] Anthony Robins. “Catastrophic forgetting, rehearsal and pseudorehearsal”. In: *Connection Science* 7.2 (1995), pp. 123–146.

[35] Olga Russakovsky et al. “Imagenet large scale visual recognition challenge”. In: *International journal of computer vision* 115 (2015), pp. 211–252.

[36] Andrei A Rusu et al. “Progressive neural networks”. In: *arXiv preprint arXiv:1606.04671* (2016).

[37] Jonathan Schwarz et al. “Progress & compress: A scalable framework for continual learning”. In: *International conference on machine learning*. PMLR. 2018, pp. 4528–4537.

[38] James Smith et al. “Always be dreaming: A new approach for data-free class-incremental learning”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021, pp. 9374–9384.

[39] James Seale Smith et al. “Coda-prompt: Continual decomposed attention-based prompting for rehearsal-free continual learning”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2023, pp. 11909–11919.

[40] Filip Szatkowski et al. “Adapt Your Teacher: Improving Knowledge Distillation for Exemplar-free Continual Learning”. In: *2023 IEEE/CVF International Conference on Computer Vision Workshops (ICCVW)* (2023), pp. 3504–3509. URL: https://api.semanticscholar.org/CorpusID:261031670.

[41] Gido M van de Ven, Tinne Tuytelaars, and Andreas S Tolias. “Three types of incremental learning”. In: *Nature Machine Intelligence* 4.12 (2022), pp. 1185–1197.

[42] Liyuan Wang et al. “A comprehensive survey of continual learning: Theory, method and application”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2024).

[43] Zifeng Wang et al. “Learning to prompt for continual learning”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022, pp. 139–149.

[44] Yue Wu et al. “Large scale incremental learning”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2019, pp. 374–382.

[45] Ye Xiang et al. “Incremental learning using conditional adversarial networks”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2019, pp. 6619–6628.

[46] Friedemann Zenke, Ben Poole, and Surya Ganguli. “Continual learning through synaptic intelligence”. In: *International conference on machine learning*. PMLR. 2017, pp. 3987–3995.

[47] G. Zhang et al. “SLCA: Slow Learner with Classifier Alignment for Continual Learning on a Pre-trained Model”. In: *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*. Los Alamitos, CA, USA: IEEE Computer Society, Oct. 2023, pp. 19091–19101. DOI: 10.1109/ICCV51070.2023.01754. URL: https://doi.ieeecomputersociety.org/10.1109/ICCV51070.2023.01754.

[48] Bowen Zhao et al. “Maintaining discrimination and fairness in class incremental learning”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2020, pp. 13208–13217.

[49] Da-Wei Zhou et al. “A Model or 603 Exemplars: Towards Memory-Efficient Class-Incremental Learning”. In: *The Eleventh International Conference on Learning Representations*. 2023. URL: https://openreview.net/forum?id=S07feAlQHgM.

[50] Fei Zhu et al. “Prototype augmentation and self-supervision for incremental learning”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2021, pp. 5871–5880.

[51] Kai Zhu et al. “Self-sustaining representation expansion for non-exemplar class-incremental learning”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022, pp. 9296–9305.

[52] Huiping Zhuang et al. “DS-AL: A Dual-Stream Analytic Learning for Exemplar-Free Class-Incremental Learning”. In: *Proceedings of the AAAI Conference on Artificial Intelligence* 38.15 (Mar. 2024), pp. 17237–17244. DOI: 10.1609/aaai.v38i15.29670. URL: https://ojs.aaai.org/index.php/AAAI/article/view/29670.

Appendix A

A.1 Base task with 50 classes

To facilitate comparison to methods using a base step of half the classes, we provide results in this setting in Table A.1. We distinguish between general continual learning (GCL) and non-GCL capable methods for fairness. We observe MITM to be a strong contender, although it is outperformed by non-GCL methods tuned for this setting.

	FAA↑	Forg↓
LUCIR [19]	41*	-
MITM	**54.52**$\pm$1.1	6.21$\pm$1.7

(a) $\mathcal{M} = 2000$

GCL	FAA↑	Forg↓
EWC [21]	21.2	-
LwF.MC [33]	27.4	-
MUC [26]	<u>30.2</u>	-
MITM	**48.78$\pm$1.07**	6.62$\pm$1.51
non-GCL		
PASS [50]	61.8	-
ABD [38]	<u>62.5</u>	-
FeTrIL [30]	**65.2**	-

(b) $\mathcal{M} = 0$

Table A.1: Final average accuracy, forgetting, S-CIFAR100 with a base task of 50 classes, and 5 classes for the subsequent 10 tasks. Results taken from FeTrIL[30]

A.2 Hyperparameter search

We have performed a hyperparameter search on S-CIFAR100 over the values given in Table A.2. Surprisingly, equivalent parameters were found for S-CIFAR10, and were also found to work well for S-TinyImageNet.

Parameter	Values
Learning rate θ	[0.1, 0.05, 0.03, **0.01**, 0.005, 0.001]
Learning rate schedule θ	[**none**, step, cosine]
Learning rate ϕ	[0.1, 0.05, 0.03, **0.01**, 0.005, 0.001]
Learning rate schedule ϕ	[**none**, step, cosine]
Epochs	[50, 75, **100**, 150, 200, 300]
Batch size	[8, 16, 24, **32**, 64, 128, 256]
Weight decay	[**0**, 1e-3, 1e-5, 1e-8]
λ	[.5, .8, **1**, 1.2, 1.5]

Table A.2: S-CIFAR10 and S-CIFAR100 parameter search. Best parameters given in bold for both $\mathcal{M} = 0$ and $\mathcal{M} = 2000$.

A.3 Cumulative setting reproduction

		Accuracy
Joint		73.31
Cumulative	$\mathcal{M} = \infty$	68.43
Experience replay	$\mathcal{M} = 2000$	38.58

Table A.3: Comparison of the Cumulative [18] and Experience replay setting to the joint training pefromance on CIFAR100. Both Cumulative and ER suffer from the stability gap as shown by [18] but it is most detrimental in the case of ER, which also observes significant class imbalance.

A.4 Pre-training the leading model

We investigate the remaining performance difference between joint training and MITM in Table A.4. We attribute a portion of this gap to the additional constraint incremental learning places over joint training. Because we do not observe all classes simultaneously, we cannot observe all features present in these samples simultaneously. We hypothesize this presents inherent limitations to the quality and generality of the learned features.

For example, when learning to separate images of zebras and cars with both classes from different tasks, we might no longer observe the stripe features of the zebra in the task with only images of the cars. When using exemplar-free distillation, this results in low variance of the features trained on detecting these stripes. As such, we cannot adequately use this feature for separation of both classes with distillation only. This inherent limitation likely bounds incremental training performance below that of joint training.

We provide some support for this claim with the following experiment. At the start of the experience, we re-initialize the leading model with CIFAR100 pre-trained weights and perform a reset of the linear layer. Optionally, every layer except the linear is frozen, to avoid unlearning of the features in the initial steps. This way, we are not dependent on the task to provide general features that are optimal with respect to the full dataset.

Joint	73.31
$\mathcal{M}$=0	
MITM pretrained + frozen	52.42
MITM pretrained	47.38
MITM	44.41
$\mathcal{M}$=2000	
MITM pretrained + frozen	63.78
MITM pretrained	63.49
MITM	56.33

Table A.4: Comparison of MITM with a pre-trained leading model on S-CIFAR100. We observe significantly better performance in the middle model if we initialize the leading model feature extractor with pre-training.