MITIGATING INTERFERENCE IN THE KNOWLEDGE CONTINUUM THROUGH ATTENTION-GUIDED INCREMENTAL LEARNING

Prashant Bhat[*,1,2], **Bharath Renjith**[*,2], **Elahe Arani**[†,1,3], **Bahram Zonooz**[†,1]
[1]Eindhoven University of Technology (TU/e) [2]TomTom [3]Wayve
{p.s.bhat, e.arani, b.zonooz}@tue.nl, bharathcrenjith@gmail.com

ABSTRACT

Continual learning (CL) remains a significant challenge for deep neural networks, as it is prone to forgetting previously acquired knowledge. Several approaches have been proposed in the literature, such as experience rehearsal, regularization, and parameter isolation, to address this problem. Although almost zero forgetting can be achieved in task-incremental learning, class-incremental learning remains highly challenging due to the problem of inter-task class separation. Limited access to previous task data makes it difficult to discriminate between classes of current and previous tasks. To address this issue, we propose 'Attention-Guided Incremental Learning' (AGILE), a novel rehearsal-based CL approach that incorporates compact task attention to effectively reduce interference between tasks. AGILE utilizes lightweight, learnable task projection vectors to transform the latent representations of a shared task attention module toward task distribution. Through extensive empirical evaluation, we show that AGILE significantly improves generalization performance by mitigating task interference and outperforming rehearsal-based approaches in several CL scenarios. Furthermore, AGILE can scale well to a large number of tasks with minimal overhead while remaining well-calibrated with reduced task-recency bias.[1]

1 INTRODUCTION

In recent years, deep neural networks (DNNs) have been shown to perform better than humans on certain specific tasks, such as Atari games (Silver et al., 2018) and classification (He et al., 2015). Although impressive, these models are trained on static data and are unable to adapt their behavior to novel tasks while maintaining performance on previous tasks when the data evolve over time (Fedus et al., 2020). Continual learning (CL) refers to a training paradigm in which DNNs are exposed to a sequence of tasks and are expected to learn potentially incrementally or online (Parisi et al., 2019). CL has remained one of the most daunting tasks for DNNs, as acquiring new information significantly deteriorates the performance of previously learned tasks, a phenomenon termed "catastrophic forgetting" (French, 1999; McCloskey & Cohen, 1989). Catastrophic forgetting arises due to the stability-plasticity dilemma (Mermillod et al., 2013), the degree to which the system must be stable to retain consolidated knowledge while also being plastic to assimilate new information. Catastrophic forgetting often results in a significant decrease in performance, and in some cases, previously learned information is completely erased by new information (Parisi et al., 2019).

Several approaches have been proposed in the literature to address the problem of catastrophic forgetting in CL. Rehearsal-based approaches (Ratcliff, 1990; Sarfraz et al., 2023; Vijayan et al., 2023) explicitly store a subset of samples from previous tasks in the memory buffer and replay them alongside current task samples to combat forgetting. In scenarios where buffer size is limited due to memory constraints (e.g., edge devices), these approaches are prone to overfitting on the buffered data (Bhat et al., 2022). On the other hand, regularization-based approaches (Kirkpatrick et al., 2017; Vijayan et al., 2023) introduce a regularization term in the optimization objective and impose a penalty for changes in parameters important for previous tasks. Although regularization greatly improves stability, these approaches often cannot discriminate classes from different tasks, thus failing miserably in scenarios such as Class-Incremental Learning (Class-IL) (Lesort et al., 2019). Parameter isolation approaches limit interference between tasks by allocating a different set of parameters for each task, either within a fixed model capacity (Gurbuz & Dovrolis, 2022; Vijayan et al., 2023) or by expanding the model size (Rusu et al., 2016; Bhat et al., 2023). However, these approaches mostly suffer from several shortcomings, including capacity saturation and scalability issues in longer task sequences. With an increasing number of tasks, selecting the right expert in the absence of task identity is nontrivial (Aljundi et al., 2017), and therefore limits its application largely to Task-Incremental Learning (Task-IL).

[*] Equal contribution. [†] Shared last author.
[1] Code: https://github.com/NeurAI-Lab/AGILE.

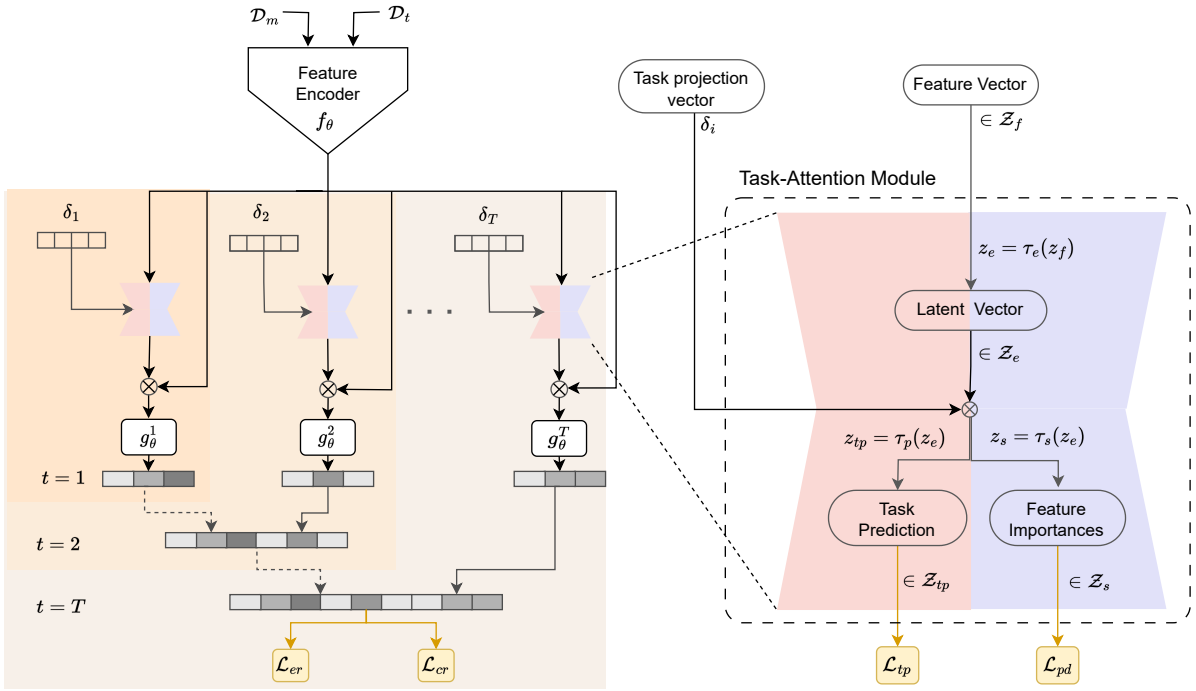

Figure 1: **A**ttention-**G**uided **I**ncremental **Le**arning (AGILE) consists of a shared task-attention module and a set of task-specific projection vectors, one for each task. Each sample is passed through the task-attention module once for each projection vector, and the outputs are fed into task-specific classifiers. AGILE effectively reduces task interference and facilitates accurate task-id prediction (TP) and within-task prediction (WP).

The problem of inter-task class separation in Class-IL remains a significant challenge due to the difficulty in establishing clear boundaries between classes of current and previous tasks (Lesort et al., 2019). When a limited number of samples from previous tasks are available in the buffer in experience rehearsal, the CL model tends to overfit on the buffered samples and incorrectly approximates the class boundaries. Kim et al. (2022) decomposes the Class-IL problem into two sub-problems: task-id prediction (TP) and within-task prediction (WP). TP involves identifying the task of a given sample, whereas WP refers to making predictions for a sample within the classes of the task identified by TP. Therefore, the Class-IL problem can be seen as a combination of the Task-IL problem (WP) and the task discovery (TP). Regardless of whether the CL algorithm defines it explicitly or implicitly, good TP and good WP are necessary and sufficient to ensure good Class-IL performance (Kim et al., 2022; Bhat et al., 2023). As task interference adversely affects both WP and TP, we hypothesize that focusing on the information relevant to the current task can facilitate more accurate TP and WP by filtering out extraneous or interfering information.

To this end, we propose 'Attention-Guided Incremental Learning' (AGILE), a rehearsal-based novel CL approach that encompasses compact task-attention to effectively mitigate interference between tasks and facilitate a good WP and TP in Class-IL. To further augment rehearsal-based learning in Class-IL, AGILE leverages parameter isolation to bring in task specificity with little computational or memory overhead. Specifically, AGILE entails a shared feature encoder and task-attention module, and as many task projection vectors as the number of tasks. Each task projection vector is a lightweight learnable vector associated with a particular task, specialized in transforming the latent representations of the shared task-attention module towards the task distribution. With dynamic expansion of task projection vectors, AGILE scales well to a large number of tasks while leaving a negligible memory footprint. Across CL scenarios, AGILE greatly reduces task interference and outperforms rehearsal-based approaches while being scalable and well-calibrated with less task-recency bias.

2 Related Works

Rehearsal-based Approaches: Earlier work sought to combat catastrophic forgetting in CL by explicitly storing and replaying previous task samples through Experience-Rehearsal (ER) (Ratcliff, 1990). Several works build on top

of ER: Since soft targets carry more information and capture complex similarity patterns in the data compared to hard targets (Hinton et al., 2015), DER++ (Buzzega et al., 2020) enforces consistency in predictions through regularization of the function space. To further improve knowledge distillation through consistency regularization, CLS-ER (Arani et al., 2022) employs multiple semantic memories that better handle the stability-plasticity trade-off. More recent works focus on reducing representation drift right after task switching to mitigate forgetting: ER-ACE (Caccia et al., 2022) through asymmetric update rules shields learned representations from drastic adaptations while accommodating new information. Co^{2}L (Cha et al., 2021) employs contrastive representation learning to learn robust features that are less susceptible to catastrophic forgetting. However, under low-buffer regimes, these approaches are prone to overfitting. Under low-buffer regimes, the quality of the buffered samples plays a significant role in defining the ability of the CL model to approximate past behavior. GCR (Tiwari et al., 2022) proposed a core set selection mechanism that approximates the gradients of the data seen so far to select and update the memory buffer. In contrast, DRI (Wang et al., 2022a) employs a generative replay to augment the memory buffer under low buffer regimes. Although reasonably successful in many CL scenarios, rehearsal-based approaches lack task-specific parameters and run the risk of shared parameters being overwritten by later tasks.

Task Attention: As the weights in DNNs hold knowledge of previous tasks, intelligent segregation of weights per task is an attractive alternative to rehearsal to reduce catastrophic forgetting in CL. Dynamic sparse parameter isolation approaches (e.g., NISPA (Gurbuz & Dovrolis, 2022), CLNP (Golkar et al., 2019), PackNet (Mallya & Lazebnik, 2018)) leverage overparameterization of DNNs and learn sparse architecture for each task within a fixed model capacity. However, these approaches suffer from capacity saturation and fail miserably in longer task sequences. By contrast, some parameter-isolation approaches grow in size, either naively or intelligently, to accommodate new tasks with the least forgetting. PNN (Rusu et al., 2016) was one of the first works to propose a growing architecture with lateral connections to previously learned features to reduce forgetting and enable forward transfer simultaneously. Since PNN instantiates a new sub-network for each task, it quickly runs into scalability issues. Approaches such as CPG (Hung et al., 2019a) and PAE (Hung et al., 2019b) grow drastically slower than PNN, but require task identity at inference. HAT (Serra et al., 2018) employed a task-based layer-wise hard attention mechanism in fully connected or convolutional networks to reduce interference between tasks. However, layer-wise attention is quite cumbersome as many low-level features can be shared across tasks. Due to the limitations mentioned above, task-specific learning approaches have been largely limited to the Task-IL setting.

Although almost zero forgetting can be achieved in Task-IL (Serra et al., 2018), the Class-IL scenario still remains highly challenging due to the problem of inter-task class separation. Therefore, we propose AGILE, a rehearsal-based CL method that encompasses task attention to facilitate a good WP and TP by reducing interference between tasks.

3 Proposed Method

3.1 Motivation

Task interference arises when multiple tasks share a common observation space but have different learning goals. In the presence of task interference, both WP and TP struggle to find the right class or task, resulting in reduced performance and higher cross-entropy loss. Continual learning in the brain is governed by the conscious processing of multiple knowledge bases anchored by a rich set of neurophysiological processes (Goyal & Bengio, 2020). Global Workspace Theory (GWT) (Baars, 1994; 2005; Baars et al., 2021) provides a formal account of cognitive information access and posits that one such knowledge base is a common representation space of fixed capacity from which information is selected, maintained, and shared with the rest of the brain (Juliani et al., 2022). During information access, the attention mechanism creates a communication bottleneck between the representation space and the global workspace, and only behaviorally relevant information is admitted into the global workspace. Such conscious processing could help the brain achieve systematic generalization (Bengio, 2017) and deal with problems that could only be solved by multiple specialized modules (VanRullen & Kanai, 2021).

In functional terms, GWT as a model of cognitive access has several benefits for CL. (i) The common representation space is largely a shared function, resulting in maximum re-usability across tasks; (ii) The attention mechanism can be interpreted as a task-specific policy for admitting task-relevant information, thereby reducing interference between tasks; And (iii) multiple specialized attention modules enable solving more complex tasks that cannot be solved by a single specialized function. Combining intuitions from both biological and theoretical findings (Appendix B), we hypothesize that focusing on the information relevant to the current task can facilitate good TP and WP, and consequently systemic generalization by filtering out extraneous or interfering information. In the following section, we describe in detail how we mitigate interference between tasks through task attention.

3.2 PRELIMINARY

Continual learning typically involves sequential tasks $t \in \{1, 2, .., T\}$ and classes $j \in \{1, 2, ..., J\}$ per task, with data appearing over time. Each task is associated with a task-specific data distribution $(\mathbf{X}_{t,j}, \mathbf{Y}_{t,j}) \in \mathcal{D}_t$. We consider two popular CL scenarios, Class-IL and Task-IL, defined in Definitions 1 and 2, respectively. Our CL model $\Phi_\theta = \{f_\theta, \tau_\theta, \delta_\theta, g_\theta\}$ consists of a backbone network (e.g. ResNet-18) f_θ, a shared attention module τ_θ, a single expanding head $g_\theta = \{g_\theta^i \mid i \leq t\}$ representing all classes for all tasks, and a set of task projection vectors up to the current task $\delta_\theta = \{\delta_i \mid i \leq t\}$.

Training DNNs sequentially has remained a daunting task since acquiring new information significantly deteriorates the performance of previously learned tasks. Therefore, to better preserve the information from previous tasks, we seek to maintain a memory buffer $\mathcal{D}_m$ that represents all tasks seen previously. We employ reservoir sampling (Algorithm 3) (Vitter, 1985) to update $\mathcal{D}_m$ throughout CL training. At each iteration, we sample a mini-batch from both $\mathcal{D}_t$ and $\mathcal{D}_m$, and update the CL model Φ_θ using experience-rehearsal as follows:

$$\mathcal{L}_{er} = \mathop{\mathbb{E}}_{(x_i,y_i)\sim\mathcal{D}_t} [\mathcal{L}_{ce}(\sigma(\Phi_\theta(x_i)), y_i)] + \alpha \mathop{\mathbb{E}}_{(x_k,y_k)\sim\mathcal{D}_m} [\mathcal{L}_{ce}(\sigma(\Phi_\theta(x_k)), y_k)] \tag{1}$$

where $\sigma(.)$ is a softmax function and $\mathcal{L}_{ce}$ is a cross-entropy loss. The learning objective for ER in Equation 1 promotes plasticity through the supervisory signal from $\mathcal{D}_t$ and improves stability through $\mathcal{D}_m$. Thus, buffer size $|D_m|$ is critical to maintaining the right balance between stability and plasticity. In scenarios where the buffer size is limited ($|D_t| \gg |D_m|$) due to memory constraints and/or privacy reasons, repeatedly learning from the restricted buffer leads to overfitting on the buffered samples. Following Arani et al. (2022), we employ an EMA of the weights ($\theta_{_{EMA}}$) of the CL model to enforce consistency in the predictions through $\mathcal{L}_{cr}$ to enable better generalization (see Appendix E.4). By implementing the complementary system theory for memory, this approach significantly enhances the model's capability to effectively retain and utilize learned information.

3.3 SHARED TASK-ATTENTION MODULE

We seek to facilitate good WP and TP by reducing task interference through task attention. Unlike multi-head self-attention in vision transformers, we propose using a shared, compact task-attention module to attend to features important for the current task. The attention module $\tau_\theta = \{\tau^e, \tau^s, \tau^{tp}\}$ consists of a feature encoder τ^e, a feature selector τ^s, and a task classifier τ^{tp}. Specifically, τ_θ is a bottleneck architecture with τ^e represented by a linear layer followed by Sigmoid activation, while τ^s is represented by another linear layer with Sigmoid activation. To orient attention to the current task, we employ a linear classifier τ^{tp} that predicts the corresponding task for a given sample.

We denote the output activation of the encoder f_θ as $z_f \in \mathbb{R}^{b\times N_f}$, τ^e as $z_e \in \mathbb{R}^{b\times N_e}$, τ^s as $z_s \in \mathbb{R}^{b\times N_s}$ and that of τ^{tp} as $z_{tp} \in \mathbb{R}^{b\times N_{tp}}$, where N_f, N_e, N_s, and N_{tp} are the dimensions of the output Euclidean spaces, and b is the batch size. To exploit task-specific features and reduce interference between tasks, we equip the attention module with a learnable task projection vector δ_i associated with each task. Each $\delta_i \in \mathbb{R}^{1\times N_e}$ is a lightweight N_e-dimensional randomly initialized vector, learnable during the corresponding task training and then fixed for the rest of the CL training. During CL training, for any sample $x \in \mathcal{D}_t \cup \mathcal{D}_m$, the incoming features z_f and the corresponding task projection vector δ_t are processed by the attention module as follows:

$$z_e = \tau^e(z_f); \quad z_s = \tau^s(z_e \otimes \delta_t); \quad z_{tp} = \tau^{tp}(z_e \otimes \delta_t). \tag{2}$$

The attention module first projects the features onto a common latent space, which is then transformed using a corresponding task projection vector. As each task is associated with a task-specific projection vector, we expect these projection vectors to capture task-specific transformation coefficients. To further encourage task-specificity in task-projection vectors, AGILE entails an auxiliary task classification:

$$\mathcal{L}_{tp} = \mathop{\mathbb{E}}_{(x,y)\sim\mathcal{D}_t} \left[\mathcal{L}_{ce}(\sigma(z_{tp}), y^t)\right] \tag{3}$$

where y^t is the ground truth of the task label.

3.4 NETWORK EXPANSION

As detailed above, the shared attention module has two inputs: the encoder output z_f and the corresponding task projection vector δ_i. As the number of tasks evolves during CL training, we propose to expand our parameter space by adding new task projection vectors commensurately. These projection vectors are sampled from a truncated normal distribution with values outside $[-2, 2]$ and redrawn until they are within the bounds. Thus, in task t there are $\{\delta_i \in$

Algorithm 1 Proposed Method: AGILE

Input: Data streams $\mathcal{D}_t$, Model $\Phi_\theta = \{f_\theta, \tau_\theta, \delta_\theta, g_\theta\}$, Hyperparameters α, β, γ, λ, Memory buffer $\mathcal{D}_m \leftarrow \{\}$
for all tasks $t \in \{1, 2, .., T\}$ **do**
  for all epochs $e \in \{1, 2, .., E\}$ **do**
    Sample a minibatch $\{x_j, y_j\}_{j=1}^{N} \in \mathcal{D}_t$
    $\hat{y}_j, z_{sj}, z_{tpj} = \text{TASKATTENTION}(x_j)$
    $\mathcal{L} = \gamma\mathcal{L}_{tp} + \lambda\mathcal{L}_{pd}$
    if $\mathcal{D}_m \neq \emptyset$ **then**
      Sample a minibatch $\{x_k, y_k\}_{k=1}^{N} \in \mathcal{D}_m$
      $\hat{y}_k, z_{sk}, z_{tpk} = \text{TASKATTENTION}(x_k)$
    $\mathcal{L}$ += $\mathcal{L}_{er} + \beta\mathcal{L}_{cr}$
    Update Φ_θ and $\mathcal{D}_m$
    Update θ_{EMA}
Return: model Φ_θ

Algorithm 2 Task-Attention

function TASKATTENTION(x):
  $z_f = f_\theta(x)$
  for all $i \leq t$ **do**
    $z_e^i = \tau^e(z_f)$
    $z_s^i = \tau^s(z_e^i \otimes \delta_i)$
    $z_{tp} = \tau^{tp}(z_e^i \otimes \delta_i)$
    $\hat{y}^i = g^i(z_s^i \otimes z_f)$
  $\hat{y}_j = concat(\hat{y}_j^i;\ \forall i \leq t)$
  return $\hat{y}, z_s, z_{tp}$

$1, 2, .., t\}$ projection vectors. For each sample, AGILE performs as many forward passes through the attention module as the number of seen tasks and generates as many feature importances ($\in \mathbb{R}^{b\times t\times N_s}$) (see Figure 1). To encourage the diversity among these feature importances, we employ a pairwise discrepancy loss as follows:

$$\mathcal{L}_{pd} = -\sum_{i=1}^{t-1} \mathop{\mathbb{E}}_{(x,y)\sim D_t} \|\sigma(z_s^t) - stopgrad(\sigma(z_s^i))\|_1 \tag{4}$$

where z_s^i is a feature importance generated with the help of the task projection vector δ_i. Since there are multiple feature importances, selecting the right feature importance is non-trivial for longer task sequences. Therefore, we propose to expand $g_\theta = \{g_\theta^i\}\ \forall\ i \leq t$ with task-specific classifiers. Each g_θ^i takes corresponding feature importance z_s^i and the encoder output z_f as input and returns predictions for classes belonging to the corresponding task. We concatenate all the outputs from task-specific classifiers and compute the final learning objective as follows:

$$\mathcal{L} = \mathcal{L}_{er} + \beta\mathcal{L}_{cr} + \gamma\mathcal{L}_{tp} + \lambda\mathcal{L}_{pd} \tag{5}$$

where β, γ, and λ are all hyperparameters. At the end of each task, we freeze the learned task projection vector and its corresponding classifier. Figure 1 depicts our proposed approach, which is detailed in Algorithms 1 and 2.

3.5 Implementation Details

We evaluate the effectiveness of our technique in two distinct CL situations: Class Incremental Learning (Class-IL) and Task Incremental Learning (Task-IL). In both Task-IL and Class-IL, each task includes a specific number of new classes that the CL model must learn. A CL model learns multiple tasks, one after the other, while being able to distinguish all the classes it has encountered so far. Task-IL is quite similar to Class-IL, with the only difference being that task labels are also provided during the inference stage, making it the simplest scenario. We obtained the popular CL datasets, Seq-CIFAR10, Seq-CIFAR100, and Seq-TinyImageNet, by dividing the original datasets CIFAR10, CIFAR100, and TinyImageNet into number tasks for the Class-IL and Task-IL scenarios: CIFAR10 into 5 tasks of 2 classes each, CIFAR100 into 5 tasks of 20 classes each, and TinyImageNet into 10 tasks of 20 classes each. In Figure 2, we divide CIFAR100 into 20 tasks of 5 classes each to compare with the parameter isolation methods. To allow for a comprehensive evaluation of different CL methods, we consider two low-buffer regimes 200 and 500, and report average accuracy on all tasks at the end of CL training.

4 Results

For all of our experiments, we employ ResNet-18 without pretraining as a backbone. Task projection vectors in AGILE are implemented as learnable parameters, while the shared task-attention module is an undercomplete autoencoder-like structure with an additional task-prediction classifier. We emphasize that we use a single expanding head so as not to be confused with a multiple-head setting. It is to remedy the problem of selecting the right task projection vector during inference. We trained all our models on NVIDIA GeForce RTX 2080 Ti (11GB). On average, it took around 2 hours to train AGILE on Seq-CIFAR10 and Seq-CIFAR100, and approximately 8 hours to train on Seq-TinyImageNet.

Table 1: Comparison of SOTA methods across various CL scenarios. We provide the average top-1 (%) accuracy of all tasks after training. † Results of the single EMA model.

Buffer	Methods	Seq-CIFAR10		Seq-CIFAR100		Seq-TinyImageNet	
		Class-IL	Task-IL	Class-IL	Task-IL	Class-IL	Task-IL
-	SGD	19.62±0.05	61.02±3.33	17.49±0.28	40.46±0.99	07.92±0.26	18.31±0.68
-	Joint	92.20±0.15	98.31±0.12	70.56±0.28	86.19±0.43	59.99±0.19	82.04±0.10
-	PNNs	-	95.13±0.72	-	74.01±1.11	-	67.84±0.29
200	ER	44.79±1.86	91.19±0.94	21.40±0.22	61.36±0.35	8.57±0.04	38.17±2.00
	DER++	64.88±1.17	91.92±0.60	29.60±1.14	62.49±1.02	10.96±1.17	40.87±1.16
	CLS-ER†	61.88±2.43	93.59±0.87	43.38±1.06	72.01±0.97	17.68±1.65	52.60±1.56
	ER-ACE	62.08±1.44	92.20±0.57	35.17±1.17	63.09±1.23	11.25± 0.54	44.17±1.02
	Co^{2}L	65.57±1.37	93.43±0.78	31.90±0.38	55.02±0.36	13.88±0.40	42.37±0.74
	GCR	64.84±1.63	90.8±1.05	33.69±1.40	64.24±0.83	13.05±0.91	42.11±1.01
	DRI	65.16±1.13	92.87±0.71	-	-	17.58±1.24	44.28±1.37
	AGILE	**69.37**±0.40	**94.25**±0.42	**45.73**±0.15	**74.37**±0.34	**20.19**±1.65	**53.47**±1.60
500	ER	57.74±0.27	93.61±0.27	28.02±0.31	68.23±0.17	9.99±0.29	48.64±0.46
	DER++	72.70±1.36	93.88±0.50	41.40±0.96	70.61±0.08	19.38±1.41	51.91±0.68
	CLS-ER†	70.40±1.21	94.35±0.38	49.97±0.78	76.37±0.12	24.97±0.80	61.57±0.63
	ER-ACE	68.45±1.78	93.47±1.00	40.67±0.06	66.45±0.71	17.73± 0.56	49.99±1.51
	Co^{2}L	74.26±0.77	**95.90**±0.26	39.21±0.39	62.98±0.58	20.12±0.42	53.04±0.69
	GCR	74.69±0.85	94.44±0.32	45.91±1.30	71.64±2.10	19.66±0.68	52.99±0.89
	DRI	72.78±1.44	93.85±0.46	-	-	22.63±0.81	52.89±0.60
	AGILE	**75.69**±0.62	95.51±0.32	**52.65**±0.93	**78.21**±0.15	**29.30**±0.53	**64.74**±0.56

4.1 Experimental results

Table 1 compares AGILE with recent rehearsal-based approaches in Class-IL and Task-IL scenarios. The associated forgetting analysis is available in Appendix 5. Several observations can be made from these results: (1) AGILE outperforms rehearsal-based approaches by a large margin across almost all datasets and buffer sizes, highlighting the importance of task attention in CL. (2) Approaches using consistency regularization (e.g., DER++ and CLS-ER) perform considerably better than others. However, AGILE demonstrates that regularization alone is insufficient to discriminate classes from different tasks. (3) While approaches addressing representation drift (e.g., Co^{2}L and ER-ACE) work well in simpler datasets, they struggle in challenging ones. For instance, in Seq-TinyImageNet with a small buffer-to-class ratio, their performance lags behind AGILE. The reliance of shared task attention on task projection vectors in AGILE helps limit representation drift by fixing these vectors after corresponding task training. (4) Approaches enhancing the quality or quantity of buffered samples (e.g., GCR and DRI) improve over vanilla ER. However, the additional computational overhead in selecting or generating buffered samples can be problematic on resource-constrained devices. In contrast, AGILE, with compact task attention using task projection vectors, outperforms rehearsal-based approaches significantly with minimal memory and computational overhead.

Task-specific learning approaches, whether within a fixed model capacity or through growth, involve isolating parameters to minimize task interference in CL. AGILE, similarly, uses task projection vectors to reduce interference between tasks. Figure 2 compares AGILE with fixed capacity models (NISPA, CLNP) and growing architectures (PNN, PAE, PackNet, CPG) trained on Seq-CIFAR100 with 20 tasks (buffer size 500 for AGILE). At the end of CL training across 20 tasks, AGILE significantly outperforms baselines with an average of 83.94%. Regarding parameter growth, PNN grows excessively, CPG grows by 1.5x, and PAE by 2x. In contrast, AGILE grows marginally by 1.01x, even for 20 tasks, without compromising performance in longer task sequences (Table 3).

4.2 How AGILE facilitates a good WP and TP?

Figure 3 (left) shows the visualization of t-distributed stochastic neighbor embedding (t-SNE) of latent features in the absence of task projection vectors. As can be seen, samples belonging to different tasks are distributed across

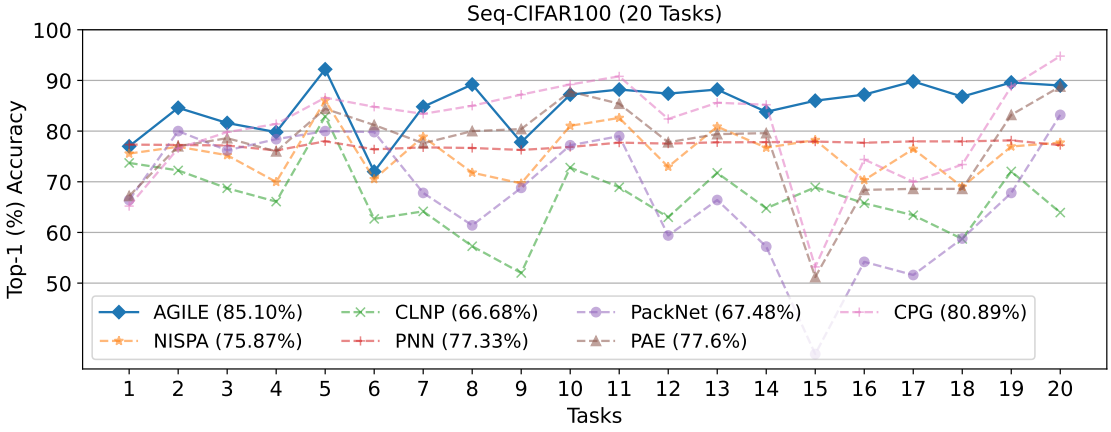

Figure 2: Comparison of AGILE with task-specific learning approaches in Task-IL setting. We report the accuracy on all tasks at the end of CL training with an average across all tasks in the legend. AGILE outperforms other baselines with little memory overhead.

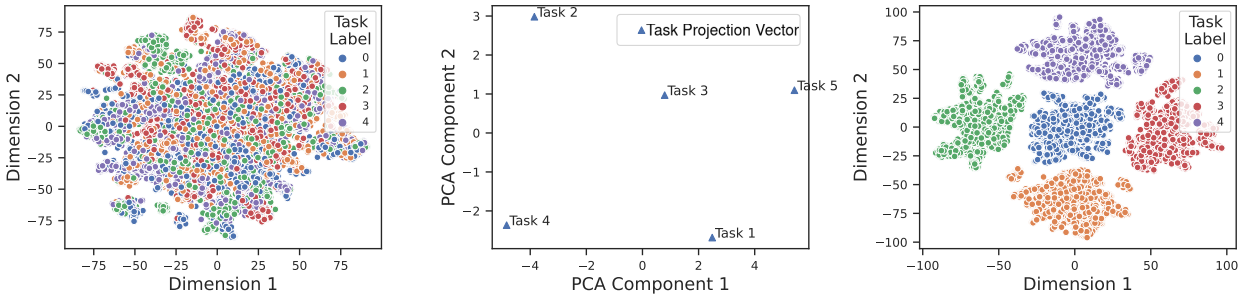

Figure 3: Latent features and task projection vectors after training on Seq-CIFAR100 with 5 tasks. (Left) t-SNE visualization of the latent features of the shared task attention module in the absence of task projection vectors; (Middle) Task projection vectors along leading principle components. (Right) t-SNE visualization of latent features of the shared task attention module in the presence of task projection vectors. Task projection vectors specialize in transforming the latent representations of shared task-attention module towards the task distribution, thereby reducing interference.

the representation space. On the other hand, Figure 3 (right) shows a t-SNE visualization of well-clustered latent features in the presence of task projection vectors. For each sample, we visualize its latent features in task attention after transforming it with the corresponding task projection vector. We also show how task projection vectors are distributed along the principal components using PCA in Figure 3 (middle). AGILE entails a shared task-attention module and as many lightweight, learnable task projection vectors as the number of tasks. As each task projection vector learns the task-specific transformation, they project samples belonging to the corresponding task differently, resulting in less interference and improved WP and TP in CL.

4.3 Ablation Study

We aim to determine the impact of each component of AGILE. As mentioned above, AGILE utilizes consistency regularization through the use of EMA and a shared task-attention mechanism with a single expanding head. Each of these components brings unique benefits to AGILE: consistency regularization aids in consolidating previous task information in scenarios with low buffer sizes, while EMA functions as an ensemble of task-specific models. Furthermore, the EMA provides better stability and acts as an inference model in our method. AGILE employs shared task-attention using task-specific projection vectors, one for each task. As the number of tasks increases, selecting the appropriate task (projection vector) without task identity becomes increasingly difficult (Aljundi et al., 2017). To address this issue, we implement a single expanding head instead of a single head, where each projection vector is responsible for classes of the corresponding task. Table 2 presents the evaluation of different components in Seq-TinyImageNet

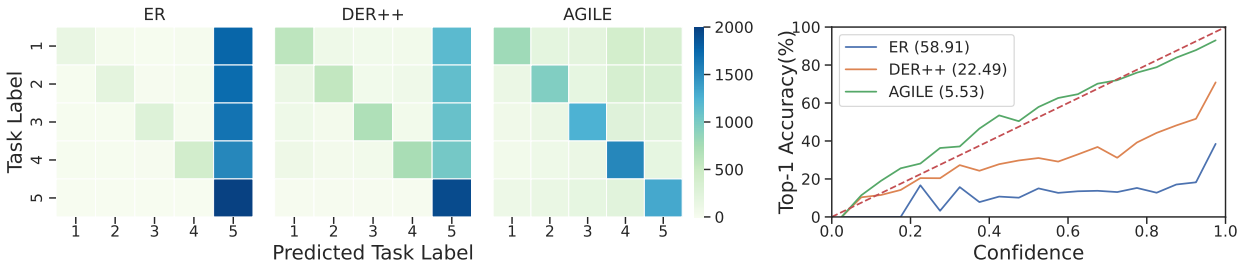

Figure 4: (Left) Confusion matrix for various CL models. ER and DER++ show high recency biases, while AGILE makes evenly distributed predictions. (Right) Reliability diagram with ECE indicating AGILE's well-calibrated performance and lowest ECE value. – denotes the perfect calibration. All models are trained on Seq-CIFAR100, 5 tasks.

(buffer size 500). As shown, AGILE takes advantage of each of these components and improves performance in both Class-IL and Task-IL settings.

Table 2: Comparison of the contributions of each of the components in AGILE. Consistency regularization in the absence of EMA implies consistency regularization by storing past logits.

Consistency regularization	EMA	Single-expanding head	Task-attention	Class-IL	Task-IL
✓	✓	✓	✓	**29.30**±0.53	**64.74**±0.56
✓	✓	✓	✗	25.43±1.07	58.89±0.84
✓	✓	✗	✗	24.97±0.80	61.57±0.63
✓	✗	✗	✗	19.38±1.41	51.91±0.68
✗	✗	✗	✗	9.99±0.29	48.64±0.46

4.4 Parameter growth

AGILE entails as many task projection vectors as the number of tasks. Therefore, the CL model grows in size as and when it encounters a new task. To this end, we compare the parameter growth in AGILE with respect to the fixed capacity model and the PNNs in Table 3. AGILE encompasses as many lightweight, learnable task projection vectors as the number of tasks, specialized in transforming the latent representations of the shared task-attention module towards the task distribution with negligible memory and computational overhead. Compared to fixed capacity models, which suffer from capacity saturation, AGILE grows marginally in size and facilitates a good within-task and task-id prediction, thereby resulting in superior performance even under longer task sequences. On the other hand, PNNs grow enormously in size, quickly rendering them unscalable in longer task sequences.

5 Model characteristics

A broader overview of the characteristics of the model is a necessary precursor for the deployment of CL in the real world. To provide a qualitative analysis, we evaluate the recency bias and model calibration for AGILE and other CL methods trained on Seq-CIFAR100 with a buffer size of 500 in Class-IL scenario.

Model Calibration: CL systems are considered well calibrated when prediction probabilities accurately reflect correctness likelihood. Despite recent high DNN accuracy, overconfident predictions (Guo et al., 2017) reduce reliability in safety-critical applications. Expected Calibration Error (ECE) estimates model reliability by gauging the difference between confidence and accuracy expectations. In Figure 4(right), different CL methods, using a calibration framework (Küppers et al., 2020), are compared. AGILE achieves the lowest ECE, showing significant calibration. By minimizing task interference, AGILE supports informed decision-making, reducing CL overconfidence.

Task Recency Bias: When a CL model learns a new task sequentially, it encounters a few samples of previous tasks while aplenty of the current task, thus skewing the learning towards the recent task (Hou et al., 2019). Ideally, the CL model is expected to have the least recent bias with predictions spread across all tasks evenly. To analyze task-recency bias, we compute the confusion matrix for different CL models .For any test sample, if the model predicts any of the classes within the sample's true task label, it is considered to have predicted the task label accurately. Figure 4 (left)

Table 3: Growth in the number of parameters (millions) for different number of task sequences in Seq-CIFAR100.

METHOD	5 TASKS	10 TASKS	20 TASKS
FIXED CAPACITY MODEL (WITH EMA)	22.461	22.461	22.461
AGILE	23.074	23.079	23.089
PNNS	297.212	874.015	2645.054

Table 4: Comparison of the forgetting measure for different CL methods for all scenarios reported in Table 1. Compared to other baselines AGILE suffers the least forgetting.

BUFFER SIZE	METHODS	SEQ-CIFAR10	SEQ-CIFAR100	SEQ-TINYIMAGENET
200	ER	61.24±2.62	75.54±0.45	76.37±0.53
	DER++	32.59±2.32	68.77±1.72	72.74±0.56
	AGILE	**25.40**±0.15	**22.74**±2.52	**36.95**±0.51
500	ER	45.35±0.07	67.74±1.29	75.27±0.17
	DER++	22.38±4.41	50.99±2.52	64.58±2.01
	AGILE	**17.57**±1.45	**22.71**±0.07	**23.97**±0.73

shows that ER and DER++ tend to predict most samples as classes in the most recent task. On the other hand, the predictions of AGILE are evenly distributed on the diagonal. Essentially, AGILE captures task-specific information through separate task projection vectors and reduces interference between tasks, resulting in the least recency bias.

Forgetting Analysis: Learning continuously on a sequence of novel tasks often results in the new information interfering with the consolidated knowledge in the model, causing catastrophic forgetting. Chaudhry et al. (2018) introduced the forgetting measure (F_T) to quantify the extent to which previously learned information is retained in the CL model. Let a_{ij} be the test accuracy of the model for task j after learning task i. Then, after training a CL model for T tasks, the forgetting measure F_T for the model is defined as,

$$F_T = \frac{1}{T-1} \sum_{t=0}^{T-1} a_t^* - a_{tT} \tag{6}$$

where a_t^* denotes the best test accuracy for the task t. After training for T tasks, a_t^* is typically computed at the task boundaries as,

$$a_t^* = \max_{l \in \{t, t+1, .., T-1\}} a_{lt}, \forall t < T \tag{7}$$

AGILE's inference model, the stochastically updated EMA, can achieve maximum accuracy for a previous task at any point during training. To compute the forgetting measure F_T, we assess AGILE on previous tasks after each epoch, comparing forgetting measures with different CL methods across datasets and buffer sizes in the Class-IL setting (see Table 4). AGILE consistently exhibits significantly lower forgetting compared to other baselines due to its task attention module and task projection vectors, which reduce interference between tasks.

6 CONCLUSION

We proposed AGILE, a novel rehearsal-based CL learning approach that employs a compact, shared task-attention module with task-specific projection vectors to effectively reduce task interference in CL. AGILE encompasses as many lightweight, learnable task projection vectors as the number of tasks, specialized in transforming the latent representations of shared task-attention module towards the task distribution with negligible memory and computational overhead. By reducing interference between tasks, AGILE facilitates good within-task and task-id prediction, resulting in superior performance across CL scenarios. With extensive empirical evaluation, we demonstrate that AGILE outperforms the rehearsal-based and parameter-isolation approaches by a large margin, signifying the efficacy of task attention in CL. Extending AGILE to rehearsal-free CL, and exploring different forms of shared task-attention are some of the useful research directions for this work.

REFERENCES

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*, pp. 3366–3375, 2017.

Elahe Arani, Fahad Sarfraz, and Bahram Zonooz. Learning fast, learning slow: A general continual learning method based on complementary learning system. In *International Conference on Learning Representations*, 2022.

Bernard J Baars. A global workspace theory of conscious experience. *Consciousness in philosophy and cognitive neuroscience*, pp. 149–171, 1994.

Bernard J Baars. Global workspace theory of consciousness: toward a cognitive neuroscience of human experience. *Progress in brain research*, 150:45–53, 2005.

Bernard J Baars, Natalie Geld, and Robert Kozma. Global workspace theory (gwt) and prefrontal cortex: Recent developments. *Frontiers in Psychology*, pp. 5163, 2021.

Jihwan Bang, Heesu Kim, YoungJoon Yoo, Jung-Woo Ha, and Jonghyun Choi. Rainbow memory: Continual learning with a memory of diverse samples. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 8218–8227, 2021.

Yoshua Bengio. The consciousness prior. *arXiv preprint arXiv:1709.08568*, 2017.

Prashant Shivaram Bhat, Bahram Zonooz, and Elahe Arani. Consistency is the key to further mitigating catastrophic forgetting in continual learning. In *Conference on Lifelong Learning Agents*, pp. 1195–1212. PMLR, 2022.

Prashant Shivaram Bhat, Bahram Zonooz, and Elahe Arani. Task-aware information routing from common representation space in lifelong learning. In *The Eleventh International Conference on Learning Representations*, 2023.

Lorenzo Bonicelli, Matteo Boschini, Angelo Porrello, Concetto Spampinato, and Simone Calderara. On the effectiveness of lipschitz-driven rehearsal in continual learning. *Advances in Neural Information Processing Systems*, 35: 31886–31901, 2022.

Matteo Boschini, Pietro Buzzega, Lorenzo Bonicelli, Angelo Porrello, and Simone Calderara. Continual semi-supervised learning through contrastive interpolation consistency. *Pattern Recognition Letters*, 162:9–14, 2022.

Pietro Buzzega, Matteo Boschini, Angelo Porrello, Davide Abati, and Simone Calderara. Dark experience for general continual learning: a strong, simple baseline. *Advances in neural information processing systems*, 33:15920–15930, 2020.

Lucas Caccia, Rahaf Aljundi, Nader Asadi, Tinne Tuytelaars, Joelle Pineau, and Eugene Belilovsky. New insights on reducing abrupt representation change in online continual learning. In *International Conference on Learning Representations*, 2022.

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

Arslan Chaudhry, Puneet K Dokania, Thalaiyasingam Ajanthan, and Philip HS Torr. Riemannian walk for incremental learning: Understanding forgetting and intransigence. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 532–547, 2018.

Hong-Jun Choi and Dong-Wan Choi. Attractive and repulsive training to address inter-task forgetting issues in continual learning. *Neurocomputing*, 500:486–498, 2022.

Arthur Douillard, Alexandre Ramé, Guillaume Couairon, and Matthieu Cord. Dytox: Transformers for continual learning with dynamic token expansion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9285–9295, 2022.

William Fedus, Dibya Ghosh, John D Martin, Marc G Bellemare, Yoshua Bengio, and Hugo Larochelle. On catastrophic interference in atari 2600 games. *arXiv preprint arXiv:2002.12499*, 2020.

Robert M French. Catastrophic forgetting in connectionist networks. *Trends in cognitive sciences*, 3(4):128–135, 1999.

Siavash Golkar, Micheal Kagan, and Kyunghyun Cho. Continual learning via neural pruning. In *Real Neurons & Hidden Units: Future directions at the intersection of neuroscience and artificial intelligence @ NeurIPS 2019*, 2019.

Anirudh Goyal and Yoshua Bengio. Inductive biases for deep learning of higher-level cognition. *arXiv preprint arXiv:2011.15091*, 2020.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. On calibration of modern neural networks. In *International conference on machine learning*, pp. 1321–1330. PMLR, 2017.

Mustafa B Gurbuz and Constantine Dovrolis. Nispa: Neuro-inspired stability-plasticity adaptation for continual learning in sparse networks. In *International Conference on Machine Learning*, pp. 8157–8174. PMLR, 2022.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pp. 1026–1034, 2015.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *stat*, 1050:9, 2015.

Saihui Hou, Xinyu Pan, Chen Change Loy, Zilei Wang, and Dahua Lin. Learning a unified classifier incrementally via rebalancing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 831–839, 2019.

Ching-Yi Hung, Cheng-Hao Tu, Cheng-En Wu, Chien-Hung Chen, Yi-Ming Chan, and Chu-Song Chen. Compacting, picking and growing for unforgetting continual learning. *Advances in Neural Information Processing Systems*, 32, 2019a.

Steven CY Hung, Jia-Hong Lee, Timmy ST Wan, Chein-Hung Chen, Yi-Ming Chan, and Chu-Song Chen. Increasingly packing multiple facial-informatics modules in a unified deep-learning model via lifelong learning. In *Proceedings of the 2019 on International Conference on Multimedia Retrieval*, pp. 339–343, 2019b.

Xisen Jin, Arka Sadhu, Junyi Du, and Xiang Ren. Gradient-based editing of memory examples for online task-free continual learning. *Advances in Neural Information Processing Systems*, 34:29193–29205, 2021.

Arthur Juliani, Kai Arulkumaran, Shuntaro Sasai, and Ryota Kanai. On the link between conscious function and general intelligence in humans and machines. *Transactions on Machine Learning Research*, 2022. ISSN 2835-8856. Survey Certification.

Gyuhak Kim, Changnan Xiao, Tatsuya Konishi, Zixuan Ke, and Bing Liu. A theoretical study on solving continual learning. *Advances in Neural Information Processing Systems*, 35:5065–5079, 2022.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017.

Fabian Küppers, Jan Kronenberger, Amirhossein Shantia, and Anselm Haselhoff. Multivariate confidence calibration for object detection. In *The IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2020.

Timothée Lesort, Andrei Stoian, and David Filliat. Regularization shortcomings for continual learning. *arXiv preprint arXiv:1912.03049*, 2019.

Arun Mallya and Svetlana Lazebnik. Packnet: Adding multiple tasks to a single network by iterative pruning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pp. 7765–7773, 2018.

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

Martial Mermillod, Aurélia Bugaiska, and Patrick Bonin. The stability-plasticity dilemma: Investigating the continuum from catastrophic forgetting to age-limited learning effects, 2013.

German I Parisi, Ronald Kemker, Jose L Part, Christopher Kanan, and Stefan Wermter. Continual lifelong learning with neural networks: A review. *Neural Networks*, 113:54–71, 2019.

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

Andrei A Rusu, Neil C Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. Progressive neural networks. *arXiv preprint arXiv:1606.04671*, 2016.

Fahad Sarfraz, Elahe Arani, and Bahram Zonooz. Synergy between synaptic consolidation and experience replay for general continual learning. In Sarath Chandar, Razvan Pascanu, and Doina Precup (eds.), *Proceedings of The 1st Conference on Lifelong Learning Agents*, volume 199 of *Proceedings of Machine Learning Research*, pp. 920–936. PMLR, 22–24 Aug 2022.

Fahad Sarfraz, Elahe Arani, and Bahram Zonooz. Error sensitivity modulation based experience replay: Mitigating abrupt representation drift in continual learning. In *The Eleventh International Conference on Learning Representations*, 2023.

Joan Serra, Didac Suris, Marius Miron, and Alexandros Karatzoglou. Overcoming catastrophic forgetting with hard attention to the task. In *International Conference on Machine Learning*, pp. 4548–4557. PMLR, 2018.

David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, et al. A general reinforcement learning algorithm that masters chess, shogi, and go through self-play. *Science*, 362(6419):1140–1144, 2018.

Rishabh Tiwari, Krishnateja Killamsetty, Rishabh Iyer, and Pradeep Shenoy. Gcr: Gradient coreset based replay buffer selection for continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 99–108, 2022.

Rufin VanRullen and Ryota Kanai. Deep learning and the global workspace theory. *Trends in Neurosciences*, 44(9): 692–704, 2021.

Eli Verwimp, Matthias De Lange, and Tinne Tuytelaars. Rehearsal revealed: The limits and merits of revisiting samples in continual learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 9385–9394, 2021.

Preetha Vijayan, Prashant Shivaram Bhat, Bahram Zonooz, and Elahe Arani. Trire: A multi-mechanism learning paradigm for continual knowledge retention and promotion. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

Jeffrey S Vitter. Random sampling with a reservoir. *ACM Transactions on Mathematical Software (TOMS)*, 11(1): 37–57, 1985.

Zhen Wang, Liu Liu, Yiqun Duan, and Dacheng Tao. Continual learning through retrieval and imagination. In *AAAI Conference on Artificial Intelligence*, volume 8, 2022a.

Zhenyi Wang, Li Shen, Le Fang, Qiuling Suo, Tiehang Duan, and Mingchen Gao. Improving task-free continual learning by distributionally robust memory evolution. In *International Conference on Machine Learning*, pp. 22985–22998. PMLR, 2022b.

Zifeng Wang, Zizhao Zhang, Chen-Yu Lee, Han Zhang, Ruoxi Sun, Xiaoqi Ren, Guolong Su, Vincent Perot, Jennifer Dy, and Tomas Pfister. Learning to prompt for continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 139–149, 2022c.

A Limitations and Future Work

We proposed AGILE to mitigate task interference and, in turn, facilitate good WP and TP through task attention. AGILE entails shared task attention and as many task projection vectors as the number of tasks. Task projection vectors capture task-specific information and are frozen after corresponding task training. Selection of the right projection vector during inference is nontrivial in longer-task sequences. To address this lacuna, we employ a single expanding head with task-specific classifiers. However, a better alternative can be developed to fully exploit task specificity in the task projection vectors. Second, AGILE strongly assumes no overlap between classes of two tasks in Class-IL / Task-IL settings. As each task-projection vector captures different information when there is non-overlap between classes, an overlap might create more confusion among projection vectors, resulting in higher forgetting. Furthermore, the shared task-attention module is still prone to forgetting due to the sequential nature of CL training.

We concentrated on integrating our approach with the ResNet18 architecture within the context of convolutional neural networks, a common practice in continual learning literature. Our method, AGILE, demonstrates superior performance compared to existing methods under uniform experimental conditions, attributed to our innovative task attention mechanism. Notably, AGILE's design, utilizing undercomplete auto-encoders and tensors without relying on architecture-specific layers like convolutions, suggests its potential applicability beyond vision tasks, including adaptation to transformer models. However, adapting AGILE to alternative architectures such as transformers involves considerable experimentation and hyperparameter adjustments, which exceeds the scope of the current discussion. We acknowledge the value of such extensions to enhance AGILE's generalizability and consider it an exciting direction for future research. Additionally, improving task-projection vector selection criterion, extending AGILE to other more complex Class-IL / Task-IL scenarios, and reducing forgetting in shared task-attention module through parameter isolation are some of the useful research directions for this work.

B Theoretical Insight

We consider a widely adopted Class-IL setting within which classes and their domains appear at most in one task, i.e., there is no overlap of classes between tasks.

Definition 1. (Class-IL): The CL model encounters $t \in \{1, 2...., T\}$ tasks with $j \in \{1, 2...., J\}$ classes per task sequentially such that the classes belonging to different tasks are disjoint i.e. for task-specific data $(\mathbf{X}_{t,j}, \mathbf{Y}_{t,j}) \in \mathcal{D}_t, \quad \mathbf{Y}_{t,j} \cap \mathbf{Y}_{t',j'} = \emptyset, \ \forall j \neq j', \ \forall t \neq t'$. Given such a setting, the primary goal of the CL model is to learn $\mathbf{P}\left(y \in \mathbf{Y}_{t,j} \mid \mathcal{D}\right)$.

For any ground event $\mathcal{D}$, Kim et al. (2022) partitioned this probability into two sub-problems, namely within-task prediction (WP) probability: $\mathbf{P}\left(y \in \mathbf{Y}_{t,j} \mid y \in \mathbf{Y}_t, D\right)$ and task-id prediction (TP) probability: $\mathbf{P}\left(y \in \mathbf{Y}_t \mid D\right)$ as follows:

$$\begin{aligned} \mathbf{P}\left(y \in \mathbf{Y}_{t_0,j_0} \mid D\right) &= \sum_{t=1,\ldots,n} \mathbf{P}\left(y \in \mathbf{Y}_{t,j_0} \mid y \in \mathbf{Y}_t, D\right) \mathbf{P}\left(y \in \mathbf{Y}_t \mid D\right) \\ &= \mathbf{P}\left(y \in \mathbf{Y}_{t_0,j_0} \mid y \in \mathbf{Y}_{t_0}, D\right) \mathbf{P}\left(y \in \mathbf{Y}_{t_0} \mid D\right) \end{aligned} \tag{8}$$

where t_0 and j_0 represent a particular task and one of its classes, respectively. TP indicates the task-id of the sample, and WP means that the prediction for a test instance is only done within the classes of the task to which the test instance belongs, which is basically the Task-IL problem as follows:

Definition 2. (Task-IL): Given the same setting as in Definition 1, the goal of the CL model is to learn the mapping function $f : \mathbf{X} \times \mathbf{T} \rightarrow \mathbf{Y}$ i.e. predict the class label $y_{j_0} \in \mathbf{Y}_{t_0}$ for a sample x from task $t_0 \in \mathbf{T}$.

In fact, it is possible to achieve almost zero forgetting in Task-IL (e.g. see Serra et al. (2018)). However, Class-IL remains challenging due to the difficulty in establishing class boundaries between classes of current and previous tasks. We seek to uncover how Class-IL performance can be further improved. To this end, Let $\mathcal{H}(p, q) = -\sum_i p_i \log q_i$ be the cross-entropy of two probability distributions p and q. We use cross-entropy as a performance measure to assess the relation between WP, TP, and Class-IL. We define the cross-entropy of WP, TP and Class-IL as $\mathcal{H}_{WP}(x) = \mathcal{H}\left(\tilde{y}, \{\mathbf{P}\left(x \in \mathbf{X}_{t_0,j} \mid x \in \mathbf{X}_{t_0}, D\right)\}_j\right)$, $\mathcal{H}_{TP}(x) = \mathcal{H}\left(\bar{y}, \{\mathbf{P}\left(x \in \mathbf{X}_t \mid D\right)\}_t\right)$ and $\mathcal{H}_{Class-IL}(x) = \mathcal{H}\left(y, \{\mathbf{P}\left(x \in \mathbf{X}_{t,j} \mid D\right)\}_{t,j}\right)$ respectively, where $\tilde{y}$, $\bar{y}$ and y are ground-truth values $\in \{0, 1\}$. We now describe how WP, TP, and Class-IL are related to each other, and how interference affects their performance.

Theorem 3. *If $\mathcal{H}_{WP}(x) \leq \epsilon$ and $\mathcal{H}_{TP}(x) \leq \xi$, then $\mathcal{H}_{Class-IL}(x) \leq \epsilon + \xi$ (Kim et al., 2022).*

For any $\epsilon > 0$ and $\xi > 0$, Theorem 3 establishes a functional relationship between WP, TP, and Class-IL. The theorem states that if $\mathcal{H}_{WP}$ and $\mathcal{H}_{TP}$ are bounded by ϵ and ξ respectively, then the Class-IL cross-entropy loss $\mathcal{H}_{Class-IL}$ is bounded by the sum of them. Therefore, having good TP and WP, lowers the upper bound of the Class-IL loss.

Task interference arises when multiple tasks share a common observation space but have different learning goals. In the presence of task interference, both WP and TP struggle to find the right class or task, resulting in reduced performance and higher cross-entropy loss. Specifically, in the presence of task interference, the upper bounds of WP and TP increase, indicating the corresponding decrease in performance, i.e. $\mathcal{H}_{WP}(x) \leq \epsilon + \hat{\epsilon}$ and $\mathcal{H}_{TP}(x) \leq \xi + \hat{\xi}$. According to Theorem 3, the upper bound of $\mathcal{H}_{Class-IL}$ will also increase proportionately. Assuming $\hat{\epsilon}, \hat{\xi} \gg 0$, task interference can have a substantial effect on overall Class-IL performance.

Therefore, it is quintessential to reduce task interference in CL to ensure optimum performance. Combining intuitions from both biological and theoretical findings, we hypothesize that focusing on the information relevant to the current task can facilitate good TP and WP, and consequently systemic generalization by filtering out extraneous or interfering information. In the following section, we describe in detail how we mitigate interference between tasks through task attention.

C Broader Related Works

In Section 2, we compared and contrasted several methods that are closely related to AGILE. We will now explore the broader related works whose problem statement overlaps with that of AGILE.

Continually learning on a sequence of tasks blurs the decision boundaries between the classes of current task and previous tasks (Lesort et al., 2019). Some approaches address inter-task forgetting indirectly by mitigating the effect of class imbalance in rehearsal-based learning. Attractive and repulsive training (ART) (Choi & Choi, 2022), aims to reduce the correlation between new and old classes through a training strategy that attracts samples from the same class while repelling other similar samples. LUCIR (Hou et al., 2019) proposes a new framework to learn a unified classifier using a combination of cosine normalization, less-forget constraint, and inter-class separation. Although these approaches aim to reduce catastrophic forgetting in CL, they address the fine-grained problem of inter-task class separation without explicitly encouraging a common representation and task-specific learning. In vision transformers, DyToX (Douillard et al., 2022) modified the final multi-head self-attention layer to act as a task attention block using task tokens. However, Dytox is an adhoc approach for transformer architectures and cannot be extended to convolutional architectures.

Although rehearsal-based approaches have been highly efficient in CL, repeated learning on a small subset of previous task data results in overfitting, thus inhibiting generalization (Verwimp et al., 2021). Several methods employ augmentation techniques, either by combining multiple data points into one (Boschini et al., 2022) or by producing multiple versions of the same buffer sample (Bang et al., 2021). Gradient-based Memory EDiting (GMED) (Jin et al., 2021) proposes editing individual examples stored in the buffer to create more challenging data to alleviate catastrophic forgetting. Distributionally Robust Optimization (DRO) (Wang et al., 2022b) proposes a principled memory evolution framework to evolve buffer data distribution focusing on population-level and distribution-level evolution. Contrary to the methods that modify memory buffer, Lipschitz Driven Rehearsal (LiDER) (Bonicelli et al., 2022) proposes a regularization objective that induces decision boundary smoothness by enforcing Lipschitz continuity of the model with respect to replay samples. Since these approaches focus on the orthogonal problem of mitigating overfitting in buffer data, they can be integrated into popular rehearsal-based approaches to further improve generalization in CL.

Learning to prompt (L2P) (Wang et al., 2022c) learns to dynamically prompt a pre-trained model to learn tasks sequentially under different task transitions. Prompts are small, learnable parameters, which are maintained in a memory space. The objective is to optimize prompts to instruct the model prediction and explicitly manage task-invariant and task-specific knowledge while maintaining model plasticity. AGILE differs from L2P in several ways: Firstly, the backbone network is learnable in AGILE, while it is fixed in L2P. As the ability to capture task-agnostic information relies heavily on the representations captured in the backbone, a fixed backbone serves the purpose only when it is pre-trained on a huge dataset. Therefore, we let the backbone learn throughout the training. Second, AGILE entails a shared feature encoder and task-attention module, and as many task projection vectors as the number of tasks. Each task projection vector is a lightweight learnable vector associated with a particular task, specialized in transforming the latent representations of the shared task-attention module towards the task distribution. On the other hand, L2P does not entail specialized task-attention modules; instead, it steers the pre-trained model toward task distribution using learnable task-specific prompts. Although similar in their dynamics, L2P aims to re-direct a pre-trained model towards a task distribution, while AGILE is a fully learnable, rehearsal-based CL approach that transforms the latent representations of the shared task-attention module towards the task distribution.

Table 5: Relative training time comparison of several CL methods trained on Seq-CIFAR100 with buffer size 200.

Method	ER	DER++	CLS-ER	AGILE
Relative time taken	1x	1.5x	1.6x	1.7x

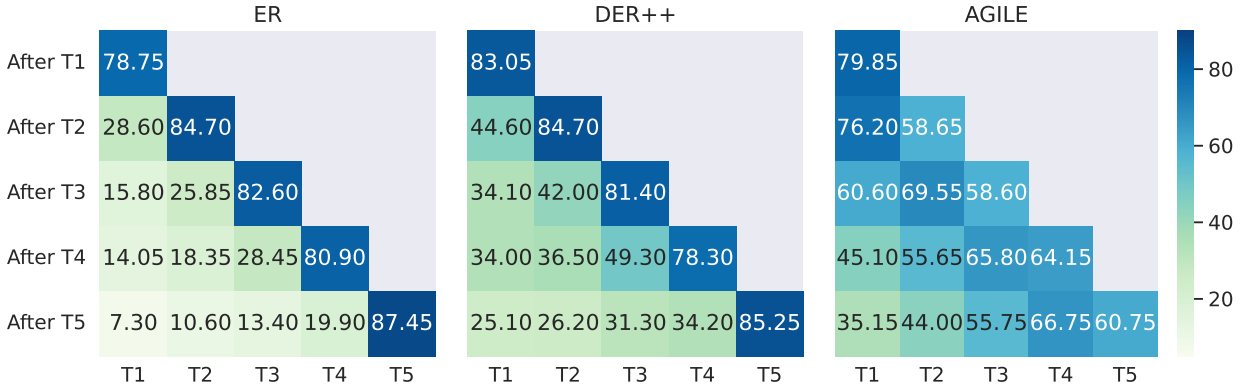

Figure 5: Task-wise performance of CL models trained on Seq-CIFAR100 with buffer size 500. The performances of ER and DER++ mainly emanate from the most recent task, while that of AGILE comes more evenly from all the tasks.

D Computational Cost

The task attention module is a lightweight undercomplete autoencoder with two MLP layers followed by ReLu activation, and the task projection vectors are implemented as learnable parameters. Thus, AGILE does not introduce any complex modules and adds negligible computational overhead. Table 5 provides the relative time taken to train on Seq-CIFAR100 for 5 tasks with a buffer size of 200. As can be seen, even with as many forward passes through the attention module as the number of tasks, the additional computational overhead incurred in AGILE is very minimal.

E Characterization of AGILE

E.1 Task-wise performance

As CL model learns new tasks in succession, it is exposed to a limited number of examples of earlier tasks, while receiving many more from the task currently being learned. This can cause the model to place more emphasis on recent tasks and less on earlier ones, leading to a bias towards the most recent tasks. Ideally, the CL model is expected to have the least recent bias with predictions spread across all tasks evenly. Figure 5 provides task-wise performance of CL models trained on Seq-CIFAR100 with buffer size 500. As can be seen, the performances of ER and DER++ emanate mostly from the final task, while that of AGILE is much more distributed across tasks.

E.2 Stability-Plasticity Dilemma

Stability of a CL model refers to its ability to retain previously learned knowledge, whereas plasticity refers to its ability to adapt to novel information. Every CL model is faced with the dilemma of finding the optimal balance between being stable and being plastic. Consequently, measuring this stability-plasticity trade-off plays a crucial role in analyzing CL models. Sarfraz et al. (2022) proposed a *Trade-off* measure that provides a formal way to estimate the balance between stability and plasticity of the model. If a CL model is trained for T tasks, then the stability ($\mathcal{S}$) of the model is defined as the average performance of all previous tasks, that is,

$$\mathcal{S} = \sum_{t=0}^{T-1} a_{Tt} \tag{9}$$

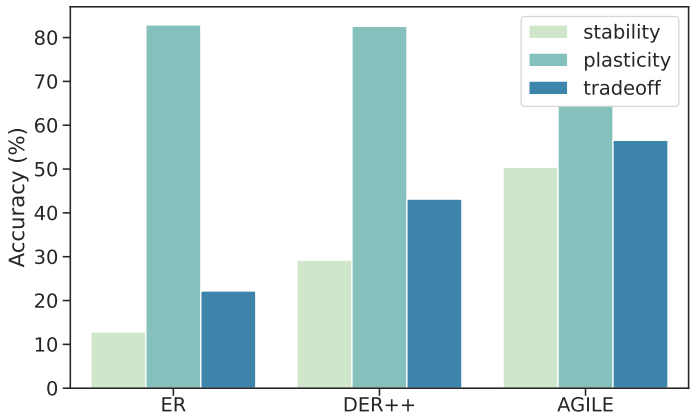

Figure 6: Stability-Plasticity Trade-off for CL models trained on Seq-CIFAR100 with 5 tasks. AGILE maintains a better balance between stability and plasticity and achieves the highest trade-off compared to other baselines.

The plasticity ($\mathcal{P}$) of the model is calculated as the average performance of learning every task for the first time, that is,

$$\mathcal{P} = \sum_{t=0}^{T} a_{tt} \tag{10}$$

The stability-plasticity trade-off is then measured as the harmonic mean of $\mathcal{S}$ and $\mathcal{P}$.

$$\textit{Trade-off} = \frac{2 \times \mathcal{S} \times \mathcal{P}}{\mathcal{S} + \mathcal{P}} \tag{11}$$

Figure 6 compares the stability, plasticity, and trade-off of different CL methods trained on Seq-CIFAR100 with 5 tasks for a buffer size of 500. While ER and DER++ quickly adapt to novel tasks, the new information interferes with the previously learned information leading to low stability. On the other hand, AGILE maintains a better balance between stability and plasticity and achieves a much higher trade-off compared to other baselines.

E.3 Reservoir sampling

We maintain a fixed size buffer $\mathcal{B}$ following the reservoir sampling strategy (Vitter, 1985). Reservoir sampling samples from a data stream of unknown length by assigning equal probability to each sample to be represented in the memory buffer. Replacements are performed at random once the buffer is full. Algorithm 3 provides the steps to maintain the buffer.

Algorithm 3 Reservoir sampling

Input: Memory buffer $\mathcal{D}_m$, maximum buffer size $\mathcal{B}$, number of seen samples N, current sample x, current label y
if $\mathcal{B} > N$ **then**
    $\mathcal{D}_m[N] \leftarrow (x, y)$
else
    $k = randomInteger(min = 0, max = N)$
    if $k < \mathcal{B}$ **then**
        $\mathcal{D}_m[k] \leftarrow (x, y)$
return $\mathcal{D}_m$

E.4 Consistency regularization using EMA

The CL model's predictions (soft-targets) capture the complex patterns and rich similarity structures in the data. As CL training progresses, soft targets (model predictions) carry more information per training sample than hard targets

Table 6: Selected hyperparameters for AGILE for all the scenarios reported in Table 1

DATASET	BUFFER SIZE	EMA PARAMS		LOSS BALANCING PARAMS				LEARNING RATE	δ_i DIMENSION
		ζ	η	α	β	γ	λ		
SEQ-CIFAR10	200	0.2	0.999	1	0.15	1	0.1	0.07	256
	500	0.2	0.999	1	0.10	1	0.1	0.05	256
SEQ-CIFAR100	200	0.05	0.999	1	0.10	1	0.1	0.03	256
	500	0.08	0.999	1	0.15	1	0.1	0.07	256
SEQ-TINYIMAGENET	200	0.05	0.999	1	0.10	1	0.1	0.05	256
	500	0.05	0.999	1	0.10	1	0.5	0.05	256

(ground truths) (Hinton et al., 2015). Therefore, in addition to ground truth labels, soft targets can be leveraged to better preserve the knowledge of the previous tasks. Consistency regularization has traditionally been used to enforce consistency in the predictions either by storing the past predictions in the buffer or by employing an exponential moving average (EMA) of the weights of the CL model. Following Arani et al. (2022), we employ an EMA of the weights of the CL model to enforce consistency in the predictions as follows:

$$\mathcal{L}_{cr} \triangleq \mathop{\mathbb{E}}_{(x_k,y_k)\sim D_m} \|\Phi_{\theta_{EMA}}(x_k) - \Phi_{\theta}(x_k)\|_F^2 \tag{12}$$

where $\|\cdot\|_F$ is the Frobenius norm, $\Phi_{\theta_{EMA}}$ is the EMA of model Φ_θ. We update the EMA model as follows:

$$\theta_{EMA} = \begin{cases} \eta\,\theta_{EMA} + (1-\eta)\,\theta, & \text{if } \zeta \geq \mathcal{U}(0,1) \\ \theta_{EMA}, & \text{otherwise} \end{cases} \tag{13}$$

where η is a decay parameter, ζ is an update rate, and θ and θ_{EMA} are the weights of the CL model and its EMA. As the knowledge of the previous tasks is encoded in the weights of the CL model, we employ EMA for inference instead of the CL model as it serves as a proxy for the self-ensemble of models specialized in different tasks.

E.5 HYPERPARAMETERS

We report the AGILE hyperparameters to reproduce the results reported in Table 1. These hyperparameters were found after tuning with multiple random initializations. In addition to these hyperparameters, we use a standard batch size of 32 and 50 epochs of training for all of our experiments. We use the SGD optimizer and other tools available in PyTorch to build AGILE.

E.5.1 HYPERPARAMETER TUNING

Table 7: Hyperparamter tuning for AGILE on Seq-CIFAR100 with buffer size 500. As can be seen, AGILE is fairly robust to choice of hyperparameters.

Varying β, for $\gamma = 1.0$, $\lambda = 0.1$		Varying γ, for $\beta = 0.15$, $\lambda = 0.15$		Varying λ, for $\beta = 0.15$, $\gamma = 1.0$	
β	Top-1 Acc %	γ	Top-1 Acc %	λ	Top-1 Acc %
0.1	52.27	0.1	51.78	**0.1**	**52.65**
0.15	**52.65**	0.5	52.46	0.2	52.33
0.2	51.98	**1.0**	**52.65**	0.5	52.31
0.5	49.56				