Continual Learning with Global Alignment

Xueying Bai **Jinghuan Shang** **Yifan Sun** **Niranjan Balasubramanian**
Department of Computer Science
Stony Brook University
{xubai, jishang, ysun, niranjan}@cs.stonybrook.edu

Abstract

Continual learning aims to sequentially learn new tasks without forgetting previous tasks' knowledge (catastrophic forgetting). One factor that can cause forgetting is the interference between the gradients on losses from different tasks. When the gradients on the current task's loss are in opposing directions to those on previous tasks' losses, updating the model for the current task may cause performance degradation on previous tasks. In this paper, we first identify causes of the above interference, and hypothesize that correlations between data representations are a key factor of interference. We then propose a method for promoting appropriate correlations between arbitrary tasks' data representations (i.e., global alignment) in individual task learning. Specifically, we learn the data representation as a task-specific composition of pre-trained token representations shared across all tasks. Then the correlations between different tasks' data representations are grounded by correlations between pre-trained token representations. We explore different ways to learn such compositions. Without experience replay, our model achieves SOTA performance in continual learning tasks. It also achieves advanced class-incremental performance through task-incremental training.

1 Introduction

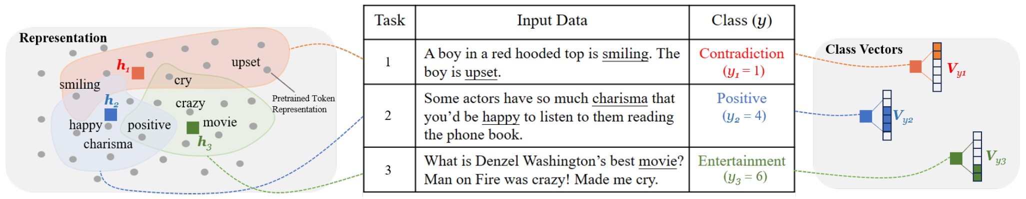
Continual Learning (CL) aims to develop models that can sequentially learn from streams of data and tasks, an important need for many real-world applications [35, 18]. One main challenge in developing CL models lies in reducing catastrophic forgetting, where models forget knowledge obtained from previous tasks after learning new tasks [38, 44, 15].

Catastrophic forgetting can happen when there is interference during task learning, especially in models that use shared parameters for all tasks [46]. Specifically, when learning a new task, if the model's gradients on the new task's loss are contradictory (e.g. in opposing directions) to those on the previous tasks' loss, the model will be updated towards a direction that increases the losses on previous tasks, causing forgetting. In this paper, we first identify factors that may lead to interference by analyzing the dot product between models' (flattened) gradients on losses from two tasks. Then, we design methods to address interference based on each factor.

Our analysis in Section 3 shows that interference mainly depends on two factors: correlations between hidden representations of the data from different tasks; and correlations between columns of the classifier (which we call class vectors) that map data representations to corresponding classes.

To address interference caused by the first factor, we want models to learn *aligned* data representations that do not have destructive correlations (i.e., leading to interference) when switching tasks and during task learning. Specifically, this requires models to accommodate future task representations when learning the current task. This motivates learning data representations based on some (well-correlated) global representations, for which we use pre-trained token representations for language data [11]. Specifically, we compose data representations as task-specific interpolations of pre-trained token

38th Conference on Neural Information Processing Systems (NeurIPS 2024).



Task	Input Data	Class (y)
1	A boy in a red hooded top is smiling. The boy is upset.	Contradiction ($y_1 = 1$)
2	Some actors have so much charisma that you'd be happy to listen to them reading the phone book.	Positive ($y_2 = 4$)
3	What is Denzel Washington's best movie? Man on Fire was crazy! Made me cry.	Entertainment ($y_3 = 6$)

Figure 1: Overview of our methods. Task i's data representations are denoted as $\mathbf{h}_i$ with pre-trained token representations as grey dots in the 'Representation' block. Correlations between aligned data representations from different tasks depends on correlations between pre-trained token representations. In the 'Class Vectors' block, class vectors for different classes have different focuses on representations after probing, which can reduce interference caused by overlapped representations.

representations. This allows the correlations between tasks' data representations to be grounded by correlations between pre-trained representations. We design three transformer-based [54] models that target such *global alignment*: (1). learning data representations by interpolating the pre-trained token representations through the attention mechanism; (2). the above model with additional neighborhood information to expand the search space of task information; (3). a controlled LoRA [20] model that adapts the pre-trained token representation with a small scaling factor.

To address interference caused by the second factor, we first train the classifier only when switching the task and then tune the whole model. This probing first strategy was first proposed in [27] to reduce representation distortion in single-task learning. Here we use it to reduce interference especially when there are destructive correlations between representations (e.g., caused by overlapping representations [5]). Probing enables different class vectors to focus on different features in data representations when switching tasks, which is useful when different tasks' representations are overlapped unexpectedly.

An overall view of our methods is in Fig. 1. Evaluations show that both the aligned representations and the probing first strategy improve CL performance in multiple settings. Specifically, global alignment models perform well in class-incremental evaluation after task-incremental training.

In conclusion, we make the following key contributions in this paper:

1. We identify factors that cause interference in CL, and propose to address the interference issues by learning aligned representations and applying the probing first strategy.
2. We design three models to learn aligned representations, which learn task-specific attention with different levels of adaptations on pre-trained token representations.
3. We conduct extensive experiments on multiple CL settings. Results show that our models can significantly reduce forgetting even without the use of experience replay.

2 Related Work

Continual Learning CL Models can be divided into three main categories: regularization-based models which constrain the deviation of new parameters from the older ones [25, 62, 1, 29]; replay-based models which reduce forgetting by rehearsing on real or pseudo samples from previous tasks [35, 7] or generative models [49, 24]; and architecture-based models which learn evolving architectures for sequential tasks, with their capacities for each task carefully assigned [48, 60].

CL in NLP is an emerging area [32, 3]. MBPA++ [10] uses experience replay and local adaptation to mitigate forgetting; LAMOL [53] generates pseudo samples for replay; IDBR [21] disentangles task-agnostic and task-specific information; CTR [23] uses a capsule network for knowledge transfer. All the above models are based on pre-trained LM [11, 4, 43]. Recent works show that pre-training can alleviate catastrophic forgetting [59, 39, 28]. Our work is also based on the pre-trained LM, but we use it for alignment purposes without experience replay.

Alignment in CL Recent CL works have studied the importance of alignment between different tasks' learning. Riemer et al. [46] aligns gradients between tasks to reduce destructive interference; Guo et al. [16] preserves holistic information for future tasks. If the previously learned knowledge does not align with that for future tasks, models may abruptly change previously learned knowledge and cause forgetting [5, 36, 22]. However, previous works focus on the alignment between observed tasks, which lack a global view and can cause forgetting in the future [26]. Instead, our work achieves alignment by pre-trained semantic features, which are general even to unseen future tasks.

Adaptation Models With limited trainable parameters, our alignment models have connections to adaptation models, which originally aimed at parameter efficiency. Different adaptation models add limited trainable parameters on the frozen transformer layer [19, 40, 17, 20]; or selectively update existing parameters [42, 61]. Recent works do adaptation by prompt tuning [31, 30, 34], which learns prompt embeddings for target tasks.

Adaptation models have also been used for CL [57, 13, 56, 45, 50]. However, most works use the models' parameter efficiency to construct progressive memory. Whether different adaptation structures influence CL, why and how they help remain unexplored. Our model has a similar form to adaptation models after derivation, but our design focuses on representation alignment rather than computational efficiency, with no progressive memory.

3 Problem Statement

In this paper, we focus on the catastrophic forgetting caused by cross-task interference [46, 47]. We conduct a case study to identify factors of the interference in Section 3, which motivates our global alignment approach in Section 4.

3.1 Continual Learning Settings

Tasks and Data We consider the setting where models continually learn a sequence of tasks, with the condition that the previous tasks' data becomes inaccessible when learning new tasks. We denote a representative data for each task i as $(\mathbf{x}_i, y_i)$, where $\mathbf{x}_i$ is the model input and $y_i \in \mathbf{c}_i$ is its class logit. $\mathbf{c}_i$ is the set of all class logits in task i, and we denote the set of class logits across all T tasks as $\mathcal{C} = \{\mathbf{c}_i\}_{i=1}^T$.

Scenarios We consider two CL scenarios: *task-incremental learning* and *class-incremental learning*. The main difference between them is that at inference time for task i the model knows its task-specific classes $\mathbf{c}_i$ in *task-incremental learning (task-aware)*, while the model has to predict the class from all classes $\mathcal{C}$ in *class-incremental learning (task-agnostic)* [37].

Models Our models consist of an encoder that encodes the input $\mathbf{x}_i$ to a d-dimensional representation $\mathbf{h}_i \in \mathbb{R}^d$, and a matrix of class vectors (i.e., a classifier) $\mathbf{v} \in \mathbb{R}^{d \times |C|}$ whose y_i-th column $\mathbf{v}_{y_i}$ maps the hidden representation $\mathbf{h}_i$ to the space of the class y_i. The probability of y_i being predicted by the model is calculated by the softmax function: $p(y_i|\mathbf{h}_i) = \text{softmax}(\mathbf{h}_i^T\mathbf{v})_{y_i}$. At training time, the softmax is computed over classes in each task, while at inference time the softmax is computed over the range of classes specified in the task- or class-incremental scenarios. We sequentially train each task with the cross-entropy loss: $\mathcal{L}(\mathbf{h}_i, y_i) = -\log p(y_i|\mathbf{h}_i)$ for task i's data $(\mathbf{x}_i, y_i)$.

3.2 Cross-Task Interference

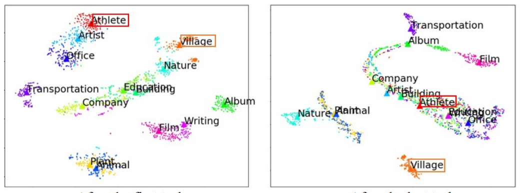
According to Riemer et al. [46], catastrophic forgetting can occur when a model learns a new task if its gradients on the new task's loss are contradictory (e.g. in opposing directions) to its gradients for the previous tasks' losses. In other words, the gradient descent for the new task might update the model towards a direction that increases its losses on previous tasks, and thus cause forgetting.

In this section, we analyze factors that can lead to interference between gradients. Specifically, we study a case based on a representative toy model. The model's encoder contains two linear layers with corresponding weight matrices denoted by $\mathbf{W}^l \in \mathbb{R}^{d \times d}$, where l indexes the layers. The encoder outputs the representation: $\mathbf{h}_i = \mathbf{W}^2\mathbf{W}^1\mathbf{x}_i$ for data in task i. Here the input $\mathbf{x}_i \in \mathbb{R}^d$ is a d-dimensional vector.

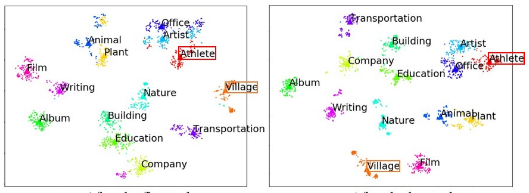
Suppose we are learning task j after task i using gradient descent. Consider $(\mathbf{x}_i, y_i)$, $(\mathbf{x}_j, y_j)$, two arbitrary data instances in tasks i and j respectively. The interference between gradients of the weight matrix $\mathbf{W}^l$ with respect to the cross-entropy loss is given by:

$$\mathcal{I}(\mathbf{W}^l) = \nabla_{\mathbf{W}^l}\mathcal{L}(\mathbf{h}_i, y_i) \cdot \nabla_{\mathbf{W}^l}\mathcal{L}(\mathbf{h}_j, y_j).$$

Destructive interference occurs when $\mathcal{I}(\mathbf{W}^l) < 0$, which can cause the model to forget task i's knowledge after updating $\mathbf{W}^l$ for task j. On the other hand, when $\mathcal{I}(\mathbf{W}^l) > 0$, the gradients of two tasks can enhance each other which encourages knowledge transfer across tasks [46].



(a) Model without global alignment



(b) Model with global alignment

Figure 2: T-SNE plots of all tasks' data representations after learning the first (with classes *Village*, *Athlete*) and last task. Under the vanilla sequential learning in (a), after the first task, representations of data from unseen tasks are overlapped. This may cause interference when switching tasks, which makes representations indistinguishable after learning the last task. With our global alignment model (Wire-Neigh) in (b), representations remain distinguishable after the first and last tasks.

We expand $\mathcal{I}(\mathbf{W}^l)$ below. For simplicity, we calculate gradients related to the y_i-th and y_j-th class vectors in the matrix $\mathbf{v}$:

$$\mathcal{I}(\mathbf{W}^l; \mathbf{v}_{y_i}, \mathbf{v}_{y_j}) = \underbrace{\big(p(y_i|\mathbf{h}_i) - 1\big)\big(p(y_j|\mathbf{h}_j) - 1\big)}_{\text{product of probability terms}\geq 0} \underbrace{(\mathbf{h}_i^{l-1})^T \mathbf{h}_j^{l-1}}_{\substack{\text{correlation between}\\ \text{hidden representations}}} \underbrace{\mathbf{v}_{y_i}^T \mathbf{\Omega}^{l+1} \mathbf{v}_{y_j}}_{\substack{\text{correlation between}\\ \text{class vectors}}} . \tag{1}$$

$\mathbf{h}_i^{l-1} = \begin{cases} \mathbf{W}^{l-1}\mathbf{x}_i, & l{=}2 \\ \mathbf{x}_i, & l{=}1 \end{cases}$ is the hidden representation at the (l-1)-th layer, $\mathbf{\Omega}^{l+1} = \begin{cases} \mathbb{I}, & l{=}2 \\ (\mathbf{W}^{l+1})^T \mathbf{W}^{l+1}, & l{=}1 \end{cases}$ is about the weight matrix at the (l+1)-th layer. The class vector $\mathbf{v}_{y_i} \in \mathbb{R}^d$ is the y_i-th column of the classifier $\mathbf{v}$, mapping the output representation to the space of class y_i.

Based on Eq. 1, $\mathcal{I}(\mathbf{W}^l)$ depends on the correlation between data's hidden representations at the (l-1)-th layer; and the correlation between class vectors transformed by weight matrices of the subsequent (e.g. (l+1)-th) layers. To address the interference issue, we discuss each correlation below.

Correlations Between Hidden Representations We consider correlations between different hidden representations at two time points (Fig. 2): (1) *The model has learned task i and is switching to learn task j*. At this time, the model's hidden representations of task j's data should not destructively interfere with those of task i's data, even though the model hasn't been trained for task j yet. (2) *After learning Task j*. At this time, the model should still produce good hidden representations for task i's data, even though it no longer has access to task i's training data.

Some previous works address the interference issue by forcing models to learn different tasks in orthogonal subspaces [9, 14, 55]. However, they may constrain models' knowledge transfer ability as $\mathcal{I}(\mathbf{W}^l)$ will always be 0. Other works minimize the destructive interference at time (2), by learning task j with the replay of task i's data [46, 47, 2]. However, if hidden representations do not correlate well when switching to task j, they may cause and propagate interference in task j's learning (Fig. 2 (a)). Then even with replay, representations of task i's data may drift towards representations of task j data, leading to disruptive model updates [5].

Our work tackles correlations of hidden representations at both time points. When learning across tasks, the model is expected to produce *aligned* hidden representations of both the current and previous tasks' data at all times. The *aligned* representations should have appropriate correlations, which will not lead to destructive interference but retain the model's ability for knowledge transfer. To achieve this, we encourage the model to learn data representations for each task as different (task-specific) interpolations of pre-trained token representations. This then enables the correlations between data representations to be grounded by correlations between pre-trained token representations (details are in Section 4).

Correlations Between Class Vectors The interference also depends on correlations between class vectors. Assuming no shared classes in task i and j, the class vector $\mathbf{v}_{y_i}$ is not involved in learning task j and thus remains unchanged after task i. When learning the class vector $\mathbf{v}_{y_j}$ for task j, we denote $\mathbf{v}_{y_j}$ at the time step t as $\mathbf{v}_{y_j,t}$. Then the correlation between class vectors $\mathbf{v}_{y_i}$ and $\mathbf{v}_{y_j,t}$ is:

$$\mathbf{v}_{y_i}^T \mathbf{v}_{y_j,t} = \mathbf{v}_{y_i}^T \mathbf{v}_{y_j,0} - \alpha \mathbf{v}_{y_i}^T \sum_t \nabla_{\mathbf{v}_{y_j}} \mathcal{L}(\mathbf{h}_{j,t}, y_j), \tag{2}$$

where $\mathbf{h}_{j,t}$ is the output data representation at time step t, $\mathbf{v}_{y_j,0}$ is the initialization of the class vector $\mathbf{v}_{y_j}$, α is the learning rate.

In Eq. 2, the correlation of class vectors depends on the initialization of the class vector $\mathbf{v}_{y_j,0}$ and the learning of data representations $\mathbf{h}_{j,t}$ (in the gradient $\nabla_{\mathbf{v}_{y_j}}\mathcal{L}(\mathbf{h}_{j,t}, y_j)$). The learning of the representation $\mathbf{h}_{j,t}$ depends on the correlations between representations, which we have discussed above. In addition, we hypothesize that a good initialization of class vectors can help mitigate the interference problem. To obtain suitable initialization for class vectors, we apply the *probing then fine-tuning* (PF) strategy [27] which first learns the class vectors (classifier) only and then fine-tune the whole model. We describe details in Section 4.

4 Methodology

In this section, we introduce our models that align data representations and initialize class vectors. First, we introduce our global alignment models which learn task data representations as interpolations of pre-trained token representations. Then we discuss the probing and then fine-tuning strategy and the effects of initializing the class vectors for CL.

4.1 Data Representation as Interpolation of Pre-Trained Token Representations

Pre-trained Token Representations In this paper, we focus on language models which are typically pre-trained in a self-supervised manner [11, 41]. For example, some models are pre-trained by the masked language modeling objective, which first masks tokens in input texts and then learns models to predict masked tokens. By pre-training, models learn semantic relationships between tokens.

We consider transformer-based [54] language models. Typically, for an arbitrary task[1], the input of the model is a sequence of n tokens. At the l-th transformer layer, denote the input representations of all tokens as $\mathbf{G}^{l-1} = [\mathbf{g}_1^{l-1}, ..., \mathbf{g}_n^{l-1}]^T$ $(l \geq 1)$ where each token representation is an $\mathbb{R}^d$ vector. Then the output token representation is:

$$\mathbf{G}^l = Attn(\mathbf{G}^{l-1}\mathbf{W}_q^l, \mathbf{G}^{l-1}\mathbf{W}_k^l)\mathbf{G}^{l-1}\mathbf{W}_v^l, \tag{3}$$

where $\mathbf{W}_q^l$ and $\mathbf{W}_k^l \in \mathbb{R}^{d\times d}$ are query and key matrices, $Attn(\mathbf{Q},\mathbf{K}) = \text{softmax}\big(\mathbf{Q}\mathbf{K}^T/\sqrt{d}\big)$ is a function calculating the attention matrix based on the query $\mathbf{Q}$ and key $\mathbf{K}$. The initial token representation $\mathbf{G}^0$ is the output of an embedding layer in the model. A feed-forward layer is applied after self-attention for token-wise transformation. We omit it here for simplicity.

After pre-training, we obtain contextual token representations $\mathbf{G}^l$ at each layer l using the pre-trained matrices $\mathbf{W}_q^l$, $\mathbf{W}_k^l$ and $\mathbf{W}_v^l$. Such token representations contain semantic information of input tokens, which are general enough to accommodate diverse uses in different downstream tasks. We refer to representations $\mathbf{G}^l$ as *pretrained token representations*.

Data Representations To address a downstream task, models learn a data representation that summarizes the task-specific information in the entire input. Typically, these data representations are learned by fine-tuning all parameters in the pre-trained model. This type of full fine-tuning has been shown to distort the pre-trained token representations [27] and may not consider correlations between data representations across tasks. This does not fit the alignment goal stated in Section 3.2.

To align data representations across tasks, we propose to wire (interpolate) the pre-trained token representations to construct the data representation. Our proposal is based on the following hypothesis:

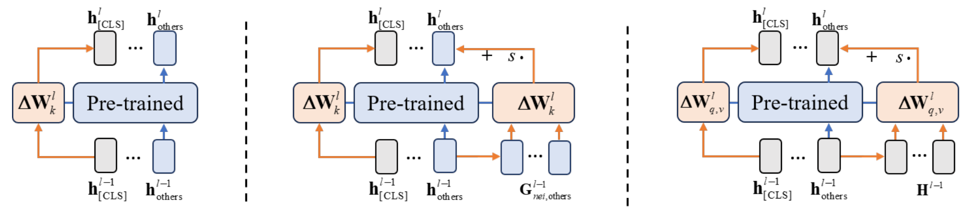
Task-specific information of data can be composed from the general semantics of tokens in that data.

For example, given the input text 'Some actors have so much <u>charisma</u> that you'd be <u>happy</u> to listen to them reading the phone book' from '*positive*' class in a sentiment analysis task, a composition of tokens {charisma, happy} can convey the information of '*positive*'.

Based on the hypothesis and the fact that pre-trained token representations contain the information of general token semantics, appropriately interpolating pre-trained token representations can represent task-specific information in the data. Since correlations between pre-trained token representations are general across tasks, this type of composition aligns data representations from different tasks.

Alignment Effect For an arbitrary task, the pre-trained token representations $\mathbf{G}^l$ can be interpolated to yield the data representation at the l-th layer as: $\mathbf{h}^l = \left(\mathbf{b}^l\mathbf{G}^l\right)^T$ where $\mathbf{b}^l \in \mathbb{R}^{1\times n}$ is a learnable

[1]For an arbitrary task, we do not specify the task id i by subscripts in the notations for simplicity.



$\mathbf{h}^l_{[\mathrm{CLS}]}$
$\mathbf{h}^l_{\mathrm{others}}$
$\Delta\mathbf{W}^l_k$
Pre-trained
$\mathbf{h}^{l-1}_{[\mathrm{CLS}]}$
$\mathbf{h}^{l-1}_{\mathrm{others}}$
$+\ s\cdot$
$\Delta\mathbf{W}^l_k$
$\mathbf{G}^{l-1}_{nei,\mathrm{others}}$
$\Delta\mathbf{W}^l_{q,v}$
$\Delta\mathbf{W}^l_{q,v}$
$\mathbf{H}^{l-1}$

(a) Fixed Wiring **(b) Wiring with Neighbor Attention** **(c) Controlled LoRA**

Figure 3: Comparison between alignment models. Modules in blue are pre-trained and in orange are learnable. Representations in grey are mainly adapted and in blue are close to the pre-trained ones. We specify hidden representations for `[CLS]` and any other token as $\mathbf{h}^l_{[\mathrm{CLS}]}$ and $\mathbf{h}^l_{\mathrm{others}}$.

stochastic row vector with weights for the interpolation. To assess the alignment effect of this scheme, we obtain the correlation between data representations $\mathbf{h}^l_i$ and $\mathbf{h}^l_j$ from task i and j as:

$$(\mathbf{h}^l_i)^T\mathbf{h}^l_j = \mathbf{b}^l_i\mathbf{G}^l_i(\mathbf{G}^l_j)^T(\mathbf{b}^l_j)^T, \tag{4}$$

where $\mathbf{b}^l_i$, $\mathbf{b}^l_j$ are learned interpolations, $\mathbf{G}^l_i$, $\mathbf{G}^l_j$ are pre-trained token representations for data in task i and j.

At any time step, the correlation between data representations is grounded by the correlation between pre-trained token representations: Eq. 4 involves the correlation $\mathbf{G}^l_i(\mathbf{G}^l_j)^T$ between pre-trained token representations; and the task-specific interpolation weights $\mathbf{b}^l_i$, $\mathbf{b}^l_j$ are also learned with the guidance of $\mathbf{G}^l_i$ and $\mathbf{G}^l_j$ respectively. This grounding to pre-trained token representations thus aligns the data representations across tasks.

4.2 Global Alignment Models

We develop global alignment models to learn data representations as interpolations of pre-trained token representations. Parameters in models are trained by the cross-entropy loss for each task. Following common practice in traditional models, we append a `[CLS]` token to the input text and use the representation of `[CLS]` as the data representation, denoted as $\mathbf{h}^l_{[\mathrm{CLS}]}$.

Fixed Wiring Applying the interpolation weights $\mathbf{b}^l$ on pre-trained token representation $\mathbf{G}^l$ generated by Eq. 3, we have $\mathbf{b}^l\mathbf{G}^l = \mathbf{b}^l Attn(\mathbf{G}^{l-1}\mathbf{W}^l_q, \mathbf{G}^{l-1}\mathbf{W}^l_k)\mathbf{G}^{l-1}\mathbf{W}^l_v$. Since the product of a stochastic row vector and a row-stochastic matrix is stochastic, $\mathbf{b}^l Attn(\mathbf{G}^{l-1}\mathbf{W}^l_q, \mathbf{G}^{l-1}\mathbf{W}^l_k)$ can be viewed as task-specific attention on the pre-trained token representations $\mathbf{G}^{l-1}$.

Based on this, we develop a *fixed wiring* model that learns task-specific attention for `[CLS]` only, while using the pre-trained parameters to compute hidden representations of other tokens as the pre-trained representations $\mathbf{G}$. In the model, the task-specific attention is calculated as the attention from `[CLS]`'s query, using a new learnable key matrix $\Delta\mathbf{W}^l_k$ and the pre-trained query matrix $\mathbf{W}^l_q$. Formally, the data representation $\mathbf{h}^l_{[\mathrm{CLS}]}$ is:

$$(\mathbf{h}^l_{[\mathrm{CLS}]})^T = Attn((\mathbf{h}^{l-1}_{[\mathrm{CLS}]})^T\mathbf{W}^l_q, \mathbf{G}^{l-1}\Delta\mathbf{W}^l_k)\mathbf{G}^{l-1}\mathbf{W}^l_v, \tag{5}$$

where $\mathbf{h}^0_{[\mathrm{CLS}]}$ is the pre-trained embedding of `[CLS]`. $\Delta\mathbf{W}^l_k$ is low-ranked [20] for efficiency.

By constraining non-cls tokens' hidden representations to be close to the pre-trained token representations, the fixed-wiring model may have limited learning capacity. To avoid this, we design two other methods with improved model capacity, which we describe below.

Wiring with Neighbor Attention Sometimes, the task information may not be easy to extract from pre-trained representations of input tokens. For example, in a text entailment task, give a sentence pair 'The boy is crying; He's happy about the view.' with the label '*contradiction*'. The pre-trained representations of task-related tokens 'crying' and 'happy' may not be negatively correlated, which makes the model hard to learn their contradiction. However, 'crying' usually has a neighbor token 'sad', and pre-trained representations of 'sad' and 'happy' are more likely to have negative correlations. Therefore, using the information of 'sad' may make the model easier to learn the task.

Therefore, to increase the model capacity while preserving its alignment ability, we retain the guidance of pre-trained token representations while exploring the tokens' neighborhood to better search for the task-specific information. The data representation can be written as:

$$(\mathbf{h}^{l}_{\texttt{[CLS]}})^T = Attn((\mathbf{h}^{l-1}_{\texttt{[CLS]}})^T\mathbf{W}^l_q, \mathbf{G}^{l-1}_{expand}\Delta\mathbf{W}^l_k)\mathbf{G}^{l-1}_{expand}\mathbf{W}^l_v, \tag{6}$$

where $\mathbf{G}_{expand} = [\mathbf{G}^{l-1}; \mathbf{G}^{l-1}_{nei}]$ concatenates input tokens' pre-trained representations $\mathbf{G}^{l-1}$ and their neighbors' representations $\mathbf{G}^{l-1}_{nei}$.

Since each token has its own neighbors, to obtain the data representation $\mathbf{h}^{l}_{\texttt{[CLS]}}$ in Eq. 6, we first adapt each pre-trained token representation individually to incorporate task-specific information from their neighbors. Then we calculate task-specific attention on adapted token representations, using the attention mechanism in Eq. 5. Specifically, we adapt the p-th pre-trained token representation $\mathbf{g}^l_p$ by:

$$(\mathbf{g}^l_p)^T \leftarrow (1-s)\cdot(\mathbf{g}^l_p)^T + s\cdot Attn((\mathbf{g}^{l-1}_p)^T\mathbf{W}^l_q, \mathbf{G}^{l-1}_{nei,p}\Delta\mathbf{W}^l_k)\mathbf{G}^{l-1}_{nei,p}\mathbf{W}^l_v,$$

where $\mathbf{G}^{l-1}_{nei,p} \in \mathbb{R}^{k\times d}$ contains k neighbor representations for the p-th token, and $\mathbf{G}^0_{nei,p}$ is the pre-trained embedding of the neighbor tokens. $s \in \mathbb{R}$ is a scaling factor. The neighbor tokens are selected by comparing cosine similarities between token embeddings. To stay close to the pre-trained token representations but incorporate task-specific information, we update the neighbor representations as: $\mathbf{G}^l_{nei,p} = (1-s)\cdot \text{expand}(\mathbf{g}^l_p) + s\cdot Attn(\mathbf{G}^{l-1}_{nei,p}\mathbf{W}^l_q, \mathbf{G}^{l-1}_{nei,p}\Delta\mathbf{W}^l_k)\mathbf{G}^{l-1}_{nei,p}\mathbf{W}^l_v$, where $\text{expand}(\mathbf{g}^l_p) \in \mathbb{R}^{k\times d}$ is the matrix which duplicates $\mathbf{g}^l_p$ for k tokens.

$\mathbf{G}_{nei}$ provides extra capacity in learning data representations. Meanwhile, the alignment effect is preserved by controlling the scale s, and making the neighborhood $\mathbf{G}_{nei}$ not deviate far away from pre-trained token representations $\mathbf{G}$.

Controlled-LoRA Another way to increase the model capacity is to adapt representations of all tokens (including both `[CLS]` and other tokens in the text) by learning low-rank matrices added to the pre-trained query and value matrices (LoRA [20]). This has the model capacity close to fine-tuning, while keeping reference to the pre-trained parameters.

Denote the input representations of all tokens at layer l as $\mathbf{H}^{l-1}$, where $\mathbf{H}^0 = \mathbf{G}^0$ is the pre-trained token embeddings. In LoRA, all token representations are updated by the same attention mechanism, with a learnable query and value matrices $\Delta\mathbf{W}^l_q$ and $\Delta\mathbf{W}^l_v$. The data representation $\mathbf{h}^{l}_{\texttt{[CLS]}}$ is:

$$(\mathbf{h}^{l}_{\texttt{[CLS]}})^T = Attn((\mathbf{h}^{l-1}_{\texttt{[CLS]}})^T(\mathbf{W}^l_q + s\cdot\Delta\mathbf{W}^l_q), \mathbf{H}^{l-1}\mathbf{W}^l_k)\mathbf{H}^{l-1}(\mathbf{W}^l_v + s\cdot\Delta\mathbf{W}^l_v). \tag{7}$$

When $s=0$, we have $\mathbf{H}^l = \mathbf{G}^l$. When $s>0$, the added query and value matrices not only learn the task-specific attention, but also adapt the token representations which can deviate away from the pre-trained ones. To keep alignment with pre-trained token representations, we control the scaling factor s to make adapted token representations close to the pre-trained ones.

4.3 Initialization of Class Vectors

Although data representations are grounded by pre-trained token representations, at the start time of learning task j after task i, the interpolation for task j's data may not be well learned. In this case, properly initializing class vectors of classes in task j can help reduce interference.

To initialize the new class vectors when switching tasks, we adopt the *probing and then fine-tuning* (PF) strategy first proposed in Kumar et al. [27]: when learning a new task, it first freezes the encoder and only trains the classifier for the task (probing); and then tunes the encoder and classifier together (fine-tuning). This is beneficial in the case, for example, when the two tasks have similar input distributions but target different classes (e.g., news sentiment analysis vs. news categorization). In this case, data representations for two tasks may overlap when switching tasks. However, the class vectors can focus on different features in data representations after probing. Therefore, the correlation between class vectors may be small (or 0) and can reduce the interference.

5 Experiments

5.1 Datasets and Metrics

We evaluate four sequences of CL tasks: (1) **Yahoo**: a split of Yahoo dataset for news question-answer categorization [63] with 5 disjoint tasks containing 2 classes each; (2) **DB**: a split of DBPedia data for

Table 1: Results for task-incremental learning using BERT-base encoder. We report the averaged accuracy (*ACC*) and forgetting (*FGT*) with their standard deviations (*std*) on five random seeds. **Bold** scores are the best scores and underline scores are the second best. ‘OOT’ means out of time.

	Model	**Yahoo**		**DB**		**News Series**		**All**	
		ACC std	*FGT std*	*ACC std*	*FGT std*	*ACC std*	*FGT std*	*ACC std*	*FGT std*
Classifier-only	Probing	$88.43_{0.06}$	—	$99.30_{0.03}$	—	$74.81_{0.46}$	—	$89.84_{0.16}$	—
Adaptation Models	FT	$73.07_{5.32}$	$18.67_{5.41}$	$73.15_{5.36}$	$24.90_{5.17}$	$59.98_{8.94}$	$21.13_{7.44}$	$60.92_{5.09}$	$30.53_{4.95}$
	Adapter	$79.85_{1.83}$	$11.86_{1.83}$	$98.70_{1.10}$	$1.19_{1.10}$	$65.43_{4.73}$	$15.53_{4.29}$	$76.31_{8.31}$	$15.97_{8.31}$
	LoRA	$86.32_{1.35}$	$5.61_{1.35}$	$88.63_{10.25}$	$11.25_{10.27}$	$69.59_{4.16}$	$12.43_{4.14}$	$77.37_{10.33}$	$14.89_{10.61}$
	Prefix	$89.75_{0.80}$	$3.04_{0.79}$	$99.83_{0.04}$	$0.07_{0.04}$	$75.03_{0.97}$	$6.13_{0.98}$	$87.53_{0.94}$	$3.80_{0.85}$
CL Models	ER	$87.42_{0.52}$	$5.61_{0.68}$	$91.05_{10.20}$	$8.70_{10.14}$	$75.47_{3.93}$	$7.81_{5.27}$	$66.42_{5.17}$	$24.91_{4.60}$
	A-GEM	$89.43_{0.58}$	$2.95_{0.64}$	$94.71_{4.70}$	$5.98_{5.49}$	$75.90_{3.34}$	$6.60_{3.84}$	$71.60_{9.38}$	$19.40_{8.15}$
	MBPA++	$86.50_{2.23}$	$5.30_{2.26}$	$97.17_{3.22}$	$2.65_{3.16}$	$72.55_{4.12}$	$7.23_{2.99}$	$82.60_{0.97}$	$9.09_{1.31}$
	IDBR (-R)	$89.32_{1.17}$	$2.19_{1.08}$	$96.47_{4.01}$	$3.39_{3.99}$	$72.36_{2.20}$	$6.50_{3.17}$	$84.09_{0.64}$	$7.00_{0.58}$
	IDBR	$90.48_{0.55}$	$1.32_{0.64}$	$99.84_{0.03}$	$0.04_{0.03}$	$76.90_{1.98}$	$3.24_{2.50}$	$88.89_{0.49}$	$2.21_{0.48}$
	CTR	$87.06_{0.98}$	$1.02_{0.74}$	$99.04_{0.81}$	$0.25_{0.30}$	$75.12_{2.32}$	$2.55_{2.19}$	OOT	—
	L2P	$90.82_{0.58}$	$0.60_{0.56}$	$99.63_{0.36}$	$0.29_{0.36}$	$73.99_{2.36}$	$3.43_{2.42}$	$87.30_{1.60}$	$3.30_{3.19}$
	CODA	$88.33_{1.12}$	$0.40_{0.34}$	$99.33_{0.23}$	$0.37_{0.25}$	$75.13_{1.02}$	$0.69_{0.74}$	$87.95_{0.33}$	$2.76_{0.38}$
Alignment Models (ours)	Wire-Fixed	$\underline{91.11}_{0.24}$	$0.70_{0.28}$	$99.85_{0.02}$	$0.03_{0.02}$	$76.28_{1.18}$	$3.66_{0.84}$	$88.63_{1.05}$	$3.55_{1.02}$
	+PF	$90.99_{0.25}$	$0.73_{0.29}$	$\underline{99.88}_{0.01}$	$0.02_{0.00}$	$77.75_{1.19}$	$1.81_{1.04}$	$\mathbf{90.18}_{0.31}$	$1.79_{0.17}$
	Wire-Neigh	$90.98_{0.28}$	$0.89_{0.36}$	$99.86_{0.01}$	$0.02_{0.01}$	$77.10_{0.99}$	$2.81_{0.52}$	$88.71_{0.31}$	$3.44_{0.51}$
	+PF	$91.10_{0.19}$	$0.63_{0.23}$	$\underline{99.88}_{0.01}$	$0.01_{0.01}$	$\underline{77.90}_{0.45}$	$1.98_{1.08}$	$\underline{89.87}_{0.37}$	$2.03_{0.33}$
	C-LoRA	$90.72_{0.89}$	$1.47_{0.97}$	$99.79_{0.05}$	$0.11_{0.05}$	$74\ .83_{2.58}$	$5.80_{2.69}$	$87.95_{0.97}$	$4.71_{1.14}$
	+PF	$\mathbf{91.38}_{0.19}$	$0.61_{0.22}$	$\mathbf{99.89}_{0.01}$	$0.01_{0.01}$	$\mathbf{78.59}_{0.85}$	$2.42_{0.94}$	$89.48_{0.64}$	$3.00_{0.83}$
Non-CL	MTL	$91.69_{0.26}$	—	$99.61_{0.41}$	—	$79.67_{1.99}$	—	$90.75_{0.46}$	—

Wikipedia article classification [63] with 7 disjoint tasks containing 2 classes each; (3) **News Series**: a sequence of tasks on news-related data, including AG_news (news classification, 4 classes), MRPC (paraphrase detection, 2 classes) [12], RTE (text entailment, 2 classes) [58] and SST (sentiment analysis, 2 classes) [51]; (4). **All**: All tasks in the above sequences. For each task, we randomly sample 1245 samples per class, which is the least number of class samples in our datasets.

We train alignment and adaptation models in task-incremental (Task-IL) settings, where in-task classes are specified during training. Then we evaluate models on both Task-IL and Class-IL inferences, where in-task classes are not specified for Class-IL inference [37]. We measure models' average accuracy and forgetting (Appendix B) over five random seeds.

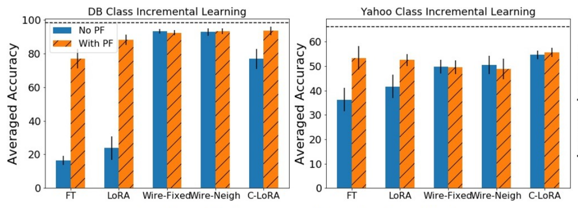
5.2 Models

We compare different models on the pre-trained BERT-base model, including:

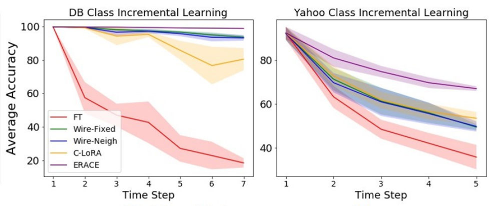
Alignment Models (ours): (1) *Wire-Fixed*: the model freezes pre-trained token representations and learns task-specific attention for data representations (i.e. `[CLS] token`). (2) *Wire-Neigh*: the wiring model with neighbor attention. We set $s = 0.1$. For computation efficiency, we fix the number of neighbors as $k = 5$, and randomly select neighbors from top-K ($K = 20$) nearest neighbors to control the range of neighborhood. (3) *C-LoRA*: the controlled LoRA model with the scaling factor $s = 0.1$. For both Wiring and C-LoRA models, we set the matrix rank $r = 8$. We also evaluate above models with the *probing then fine-tuning* (PF) strategy, denoted as *Model+PF*.

Adaptation Models: (1) *Fine-tuning (FT)*: fine-tuning all parameters sequentially. (2) *Prefix Tuning (Prefix)* [33]: freezing the pre-trained parameters and adding learnable embeddings to attention layers. (3) *Adapter* [19]: freezing the pre-trained parameters and injecting learnable linear projections after self-attention. (4) *LoRA* [20]: the LoRA model with suggested scaling $s = 1$ for single task learning.

CL Models: (1) *ER*: the FT model storing all seen examples and performs sparse (1%) experience replay. (2) *A-GEM* [7]: the FT model constraining gradients to prevent degrading performance on previous tasks. (3) *MBPA++* [10]: the FT model that stores and retrieves samples to locally adapt the model at inference time like [52]. (4) *IDBR* [21]: the FT model with information-disentanglement-based regularization and replay. We also compare to IDBR without replay, denoted as *IDBR(-R)*. (5) *CTR* [23]: an adapter-based model with capsules and task transfer routing. (6) *L2P* [57]: a prompt-based model that dynamically prompts for different data and tasks. (7) *CODA* [50]: a prompt-based model that learns attention over extensive prompt components for tasks. (8). *ERACE* [5]: a model for class-IL that calculates the current task's loss over in-task classes, while calculating the replay loss over all seen classes in the replay buffer. We also show the performance of *MTL*, which is an FT model jointly trained on all tasks (not CL). Detailed settings for models are shown in Appendix A.



(a) Class-IL Accuracy After the Last Task



(b) Class-IL Accuracy across Time

Figure 4: (a). Class-IL accuracy after the last task. Dashed lines show accuracies of ERACE, which replays previous tasks' data with Class-IL loss. (b). Average Class-IL accuracies after each task.

Table 2: Tokens decoded from data representations by the pre-trained MLM decoder. Red tokens are tokens related to target classes in inputs; blue tokens are those related to red tokens in decodings.

Data (Task 1)	*Task*	**Yahoo (news categorization, target class: Science)**	**DB (article classification, target class: Film)**
	Sentence 1	what is the meaning of blog?	No One Man
	Sentence 2	is a little space of paper of webpage space where u can write a little sentence expressing a general idea that represents what u feel or think.	No One Man is a 1932 American drama film starring Carole Lombard and Ricardo Cortez and directed by Lloyd Corrigan. It is based on a novel by Rupert Hughes.
Top-10 Decoded Tokens of [CLS]	FT	1 A One a h An an 2 x and	. the ; : all its , his their as
	LoRA	Every ##nt End then X : end E G .	from to pro Sc card into B in stick long
	Wire-Fixed	blog ##ly user ##log Facebook blogs ##ing : Twitter web	and released , . - (directed made / Western
	Wire-Neigh	blog post site posting @ web page content crawl a	. and directed is , 2 film - produced released
	C-LoRA	. : content blog Wikipedia posting ; # post page	is , film . of and are but - follows

5.3 Results

Alignment models are effective in Task-IL and Class-IL without replay. Results for Task-IL are shown in Table 1. Our alignment models forget less than CL models that use experience replay during fine-tuning. They also outperform adaptation models including Adapter, uncontrolled LoRA and prefix tuning. This suggests the reduced forgetting in our alignment models is not just due to tuning fewer parameters, a feature in common with the adaptation models. In particular, C-LoRA performs better than the original LoRA model, which does not control the scaling factor for alignment. Results for Class-IL are shown in Fig. 4. On DB, alignment models achieve advanced Class-IL performance close to ERACE, a strong Class-IL model with replay, even under the Task-IL training where models are not trained on out-task classes. On Yahoo, alignment models also forget less than FT and LoRA.

Wiring models forget less. Without the probing and fine-tuning (PF) strategy, wiring models achieve best performances in all Task-IL experiments and Class-IL on DB (Fig. 4). We hypothesize that the good performance of wiring models is due to the strong alignment effect achieved by referencing the pre-trained token representations. With improved capacity brought by neighbors, Wire-Neigh outperforms Wire-Fixed on hard tasks like those in News Series. However, wiring models may have less capacity compared to models with more adaptations, which can limit their classification (separation) capability in Class-IL (e.g. Class-IL on Yahoo).

PF reduces forgetting. Fig. 4 shows that applying the PF strategy improves CL performance for all models. Although C-LoRA itself performs worse than wiring models in Task-IL, with PF it outperforms wiring models on 3/4 sequences. It also achieves best performance on Yahoo Class-IL. We hypothesize that is because PF can reduce interference when data representations have destructive correlations, as discussed in Section 3.2. Generally, PF and alignment on representations compensate each other on reducing interference. When the alignment effect is strong (e.g., wiring models), the improvement caused by PF is not very significant. However, when the alignment effect is not as strong (e.g., C-LoRA, LoRA, FT), PF can significantly improve models' performance in CL.

5.4 Additional Analysis and Ablations

Interpretability of Alignment Models Since alignment models learn data representations as the interpolation of pre-trained token representations, we interpret learned data representations using the pre-trained MLM decoder. Specifically, we decode data representations to the token space by the pre-trained decoder and then get the tokens with top-10 probabilities. This helps us understand how

Table 3: Alignment quantification on SNLI and News Series. We report Recall@20 on three random seeds.

Model	SNLI	News Series
FT	6.80 ± 1.72	7.74 ± 3.3
C-LoRA	14.53 ± 0.63	23.13 ± 2.75
Wire-Fixed	**37.01** ± 1.54	27.80 ± 1.48
Wire-Neigh	36.24 ± 1.83	**32.32**± 2.58

data representations are closed to pre-trained token representations the model aligns to. Results are shown in Table 2. From the table, decoded tokens of alignment models are close to tokens that are related to target classes, while those of non-alignment models (FT, LoRA) are hard to interpret.

We also quantify the model's alignment ability using E-SNLI data [6], where each data's task-related tokens are annotated by human. We calculate the Recall@20 of annotated task-related tokens being retrieved from data representations, on SNLI and News Series data. The results are shown in Table 3. Results suggest that wiring models have more alignment ability than C-LoRA, in both in-task (SNLI) and CL evaluations on similar NLI tasks (News Series).

This interpretability may explain the effectiveness of alignment models in Class-IL: even when trained with local classes, data representations in each task are correlated to pre-trained token representations that relate to all tasks' classes. This may help to separate representations from different tasks.

Influence of Scaling Factor We show Task-IL accuracies of different scaling factors s for Wire-Neigh and C-LoRA in Table 4. C-LoRA's CL performance tends to decrease when s increases. This may be due to the decrease of the global alignment effect, which increases the interference. After applying PF, C-LoRA's accuracy first increases and then slightly decreases. This may be because PF reduces the interference caused by the class vectors, and the model can fully utilize its global alignment ability when increasing plasticity. However, when the scaling factor goes too large, the loss of alignment will lead to more forgetting even with PF. For Wire-Neigh, the observation is similar to C-LoRA: when s goes up, the model's accuracy first increases and then decreases because of the trade-offs between global alignment and plasticity. And since Wire-Neigh interpolate pre-trained token representations with their neighbor representations, the increase of s leads to the decrease of pre-trained information. And therefore we observe a more rapid performance drop when s increases.

Table 4: Average *ACC* with different s in C-LoRA, Wire-Neigh on News Series.

Model	$s = 0$	0.1	0.4	0.7	1.0
Wire-Neigh	76.28	77.10	72.59	68.18	66.59
C-LoRA	74.81	74.83	72.99	71.02	69.59
+PF	74.81	78.59	77.41	76.83	76.81

Influence of Neighborhood In Wire-Neigh, we randomly select five neighbor tokens from top-K nearest neighbors. Here we study the effect of the range of neighborhood with different K value in Table 5. For relatively simple sequences DB and Yahoo, Wire-Neigh under different K has stable performance. However, for hard sequence News Series, when K increases, the model has more neighbor information (more capacity) to solve the task, which first improves its CL performance. However, when K is too large ($K = 100$), the neighbor information may become noisy, which makes the CL performance drop.

Table 5: Average *ACC* with different neighborhood range K in Wire-Neigh.

K	**Yahoo**	**DB**	**News Series**
5	99.86	91.16	76.90
20	99.86	90.98	77.10
50	99.86	91.16	77.20
100	99.87	91.13	76.58

6 Limitation

We discuss our limitations in model and assumption perspectives: For models, without replaying previous data, there can be a problem of shifting attention; e.g. the model shifts attention on previous task-related tokens after learning new tasks. This may lead to forgetting, which we leave as our future study. For assumptions, our model assumes we have a pre-trained model. For domains that do not have well-established pre-trained semantic features, our model may not be immediately applicable. However, since fundamental models are consistently established and shown to be beneficial across different domains, we can expect our models to apply to more domains in the future.

7 Conclusion

In this paper, we investigate methods to address correlations between data representations and class vectors to reduce interference when training across tasks. Specifically, for alignment, we propose to learn data representations as task-specific compositions of pre-trained token representations. To learn the composed representations, we propose wiring models with or without neighbor attention and a controlled LoRA model. To address correlations between class vectors, we adopt the probing and then fine-tuning strategy, which can effectively reduce interference even when the representations do not correlate well. Experiments show that our models can successfully learn the composed representations for alignment and achieve SOTA performance in CL.

Acknowledgments

We thank the anonymous reviewers for their insightful feedback to improve the paper. This material is based on research that is supported in part by the Air Force Research Laboratory (AFRL), DARPA, for the KAIROS program under agreement number FA8750-19-2-1003 and in part by the National Science Foundation under the award IIS #2007290.

References

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

[2] Rahaf Aljundi, Eugene Belilovsky, Tinne Tuytelaars, Laurent Charlin, Massimo Caccia, Min Lin, and Lucas Page-Caccia. Online continual learning with maximal interfered retrieval. *Advances in neural information processing systems*, 32, 2019.

[3] Magdalena Biesialska, Katarzyna Biesialska, and Marta R. Costa-jussà. Continual lifelong learning in natural language processing: A survey. In *Proceedings of the 28th International Conference on Computational Linguistics*, pp. 6523–6541, Barcelona, Spain (Online), December 2020. International Committee on Computational Linguistics. doi: 10.18653/v1/2020.coling-main.574. URL https://www.aclweb.org/anthology/2020.coling-main.574.

[4] Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*, 2020.

[5] 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. URL https://openreview.net/forum?id=N8MaByOzUfb.

[6] Oana-Maria Camburu, Tim Rocktäschel, Thomas Lukasiewicz, and Phil Blunsom. e-snli: Natural language inference with natural language explanations. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

[7] Arslan Chaudhry, Marc'Aurelio Ranzato, Marcus Rohrbach, and Mohamed Elhoseiny. Efficient lifelong learning with a-GEM. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=Hkf2_sC5FX.

[8] Arslan Chaudhry, Marcus Rohrbach, Mohamed Elhoseiny, Thalaiyasingam Ajanthan, P Dokania, P Torr, and M Ranzato. Continual learning with tiny episodic memories. In *Workshop on Multi-Task and Lifelong Reinforcement Learning*, 2019.

[9] Arslan Chaudhry, Naeemullah Khan, Puneet Dokania, and Philip Torr. Continual learning in low-rank orthogonal subspaces. *Advances in Neural Information Processing Systems*, 33: 9900–9911, 2020.

[10] Cyprien de Masson D'Autume, Sebastian Ruder, Lingpeng Kong, and Dani Yogatama. Episodic memory in lifelong language learning. *Advances in Neural Information Processing Systems*, 32, 2019.

[11] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, 2019.

[12] Bill Dolan and Chris Brockett. Automatically constructing a corpus of sentential paraphrases. In *Third International Workshop on Paraphrasing (IWP2005)*, 2005.

[13] Beyza Ermis, Giovanni Zappella, Martin Wistuba, Aditya Rawal, and Cedric Archambeau. Memory efficient continual learning with transformers. *Advances in Neural Information Processing Systems*, 35:10629–10642, 2022.

[14] Mehrdad Farajtabar, Navid Azizan, Alex Mott, and Ang Li. Orthogonal gradient descent for continual learning. In *International Conference on Artificial Intelligence and Statistics*, pp. 3762–3773. PMLR, 2020.

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

[16] Yiduo Guo, Bing Liu, and Dongyan Zhao. Online continual learning through mutual information maximization. In *International Conference on Machine Learning*, pp. 8109–8126. PMLR, 2022.

[17] Junxian He, Chunting Zhou, Xuezhe Ma, Taylor Berg-Kirkpatrick, and Graham Neubig. Towards a unified view of parameter-efficient transfer learning. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=0RDcd5Axok.

[18] 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.

[19] Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for nlp. In *International Conference on Machine Learning*, pp. 2790–2799. PMLR, 2019.

[20] Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9.

[21] Yufan Huang, Yanzhe Zhang, Jiaao Chen, Xuezhi Wang, and Diyi Yang. Continual learning for text classification with information disentanglement based regularization. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2021.

[22] Khurram Javed and Martha White. Meta-learning representations for continual learning. *Advances in Neural Information Processing Systems*, 32, 2019.

[23] Zixuan Ke, Bing Liu, Nianzu Ma, Hu Xu, and Lei Shu. Achieving forgetting prevention and knowledge transfer in continual learning. *Advances in Neural Information Processing Systems*, 34:22443–22456, 2021.

[24] Ronald Kemker and Christopher Kanan. Fearnet: Brain-inspired model for incremental learning. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=SJ1Xmf-Rb.

[25] 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.

[26] Jeremias Knoblauch, Hisham Husain, and Tom Diethe. Optimal continual learning has perfect memory and is NP-hard. In Hal Daumé III and Aarti Singh (eds.), *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 5327–5337. PMLR, 13–18 Jul 2020. URL https://proceedings.mlr.press/v119/knoblauch20a.html.

[27] Ananya Kumar, Aditi Raghunathan, Robbie Matthew Jones, Tengyu Ma, and Percy Liang. Fine-tuning can distort pretrained features and underperform out-of-distribution. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=UYneFzXSJWh.

[28] Kuan-Ying Lee, Yuanyi Zhong, and Yu-Xiong Wang. Do pre-trained models benefit equally in continual learning? In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pp. 6485–6493, January 2023.

[29] Sang-Woo Lee, Jin-Hwa Kim, Jaehyun Jun, Jung-Woo Ha, and Byoung-Tak Zhang. Overcoming catastrophic forgetting by incremental moment matching. *Advances in neural information processing systems*, 30, 2017.

[30] Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, 2021.

[31] Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, 2021.

[32] Tianlin Liu, Lyle Ungar, and Joao Sedoc. Continual learning for sentence representations using conceptors. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, 2019.

[33] Xiao Liu, Kaixuan Ji, Yicheng Fu, Zhengxiao Du, Zhilin Yang, and Jie Tang. P-tuning v2: Prompt tuning can be comparable to fine-tuning universally across scales and tasks. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, 2021.

[34] Xiao Liu, Yanan Zheng, Zhengxiao Du, Ming Ding, Yujie Qian, Zhilin Yang, and Jie Tang. Gpt understands, too. *arXiv preprint arXiv:2103.10385*, 2021.

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

[36] Divyam Madaan, Jaehong Yoon, Yuanchun Li, Yunxin Liu, and Sung Ju Hwang. Representational continuity for unsupervised continual learning. In *International Conference on Learning Representations*, 2021.

[37] Marc Masana, Xialei Liu, Bartłomiej Twardowski, Mikel Menta, Andrew D Bagdanov, and Joost van de Weijer. Class-incremental learning: survey and performance evaluation on image classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022.

[38] 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.

[39] Sanket Vaibhav Mehta, Darshan Patil, Sarath Chandar, and Emma Strubell. An empirical investigation of the role of pre-training in lifelong learning. *Journal of Machine Learning Research*, 24(214):1–50, 2023.

[40] Jonas Pfeiffer, Aishwarya Kamath, Andreas Rücklé, Kyunghyun Cho, and Iryna Gurevych. AdapterFusion: Non-destructive task composition for transfer learning. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pp. 487–503, Online, April 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.eacl-main.39.

[41] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training. *https://s3-us-west-2. amazonaws. com/openai-assets/researchcovers/languageunsupervised/language understanding paper. pdf*, 2018.

[42] Evani Radiya-Dixit and Xin Wang. How fine can fine-tuning be? learning efficient language models. In Silvia Chiappa and Roberto Calandra (eds.), *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, volume 108 of *Proceedings of Machine Learning Research*, pp. 2435–2443. PMLR, 26–28 Aug 2020.

[43] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67, 2020.

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

[45] Anastasia Razdaibiedina, Yuning Mao, Rui Hou, Madian Khabsa, Mike Lewis, and Amjad Almahairi. Progressive prompts: Continual learning for language models. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=UJTgQBc91_.

[46] Matthew Riemer, Ignacio Cases, Robert Ajemian, Miao Liu, Irina Rish, Yuhai Tu, , and Gerald Tesauro. Learning to learn without forgetting by maximizing transfer and minimizing interference. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=B1gTShAct7.

[47] David Rolnick, Arun Ahuja, Jonathan Schwarz, Timothy Lillicrap, and Gregory Wayne. Experience replay for continual learning. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/file/fa7cdfad1a5aaf8370ebeda47a1ff1c3-Paper.pdf.

[48] 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.

[49] Hanul Shin, Jung Kwon Lee, Jaehong Kim, and Jiwon Kim. Continual learning with deep generative replay. *Advances in neural information processing systems*, 30, 2017.

[50] James Seale Smith, Leonid Karlinsky, Vyshnavi Gutta, Paola Cascante-Bonilla, Donghyun Kim, Assaf Arbelle, Rameswar Panda, Rogerio Feris, and Zsolt Kira. 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*, pp. 11909–11919, 2023.

[51] Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pp. 1631–1642, 2013.

[52] Pablo Sprechmann, Siddhant Jayakumar, Jack Rae, Alexander Pritzel, Adria Puigdomenech Badia, Benigno Uria, Oriol Vinyals, Demis Hassabis, Razvan Pascanu, and Charles Blundell. Memory-based parameter adaptation. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=rkfOvGbCW.

[53] Fan-Keng Sun, Cheng-Hao Ho, and Hung-Yi Lee. Lamol: Language modeling for lifelong language learning. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=Skgxcn4YDS.

[54] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pp. 5998–6008, 2017.

[55] Xiao Wang, Tianze Chen, Qiming Ge, Han Xia, Rong Bao, Rui Zheng, Qi Zhang, Tao Gui, and Xuanjing Huang. Orthogonal subspace learning for language model continual learning. In *The 2023 Conference on Empirical Methods in Natural Language Processing*, 2023. URL https://openreview.net/forum?id=L7ZBpZZ8Va.

[56] Zifeng Wang, Zizhao Zhang, Sayna Ebrahimi, Ruoxi Sun, Han Zhang, Chen-Yu Lee, Xiaoqi Ren, Guolong Su, Vincent Perot, Jennifer Dy, et al. Dualprompt: Complementary prompting for rehearsal-free continual learning. In *European Conference on Computer Vision*, pp. 631–648. Springer, 2022.

[57] 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, 2022.

[58] Adina Williams, Nikita Nangia, and Samuel Bowman. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pp. 1112–1122, New Orleans, Louisiana, June 2018. Association for Computational Linguistics. doi: 10.18653/v1/N18-1101. URL https://www.aclweb.org/anthology/N18-1101.

[59] Tongtong Wu, Massimo Caccia, Zhuang Li, Yuan-Fang Li, Guilin Qi, and Gholamreza Haffari. Pretrained language model in continual learning: A comparative study. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=figzpGMrdD.

[60] Jaehong Yoon, Eunho Yang, Jeongtae Lee, and Sung Ju Hwang. Lifelong learning with dynamically expandable networks. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=Sk7KsfW0-.

[61] Elad Ben Zaken, Shauli Ravfogel, and Yoav Goldberg. Bitfit: Simple parameter-efficient fine-tuning for transformer-based masked language-models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, 2021.

[62] Friedemann Zenke, Ben Poole, and Surya Ganguli. Continual learning through synaptic intelligence. In *International Conference on Machine Learning*, pp. 3987–3995. PMLR, 2017.

[63] Xiang Zhang, Junbo Zhao, and Yann LeCun. Character-level convolutional networks for text classification. *Advances in neural information processing systems*, 28, 2015.

A Detailed Experimental Settings

We provide detailed experimental settings in addition to the main paper Section 5. We train all models (except ERACE) via Task-IL training, while evaluating them in both Task-IL and Class-IL settings. We perform all experiments on one Nvidia RTX A6000 machine.

- **Probing**: We fix the encoder and only train the classifier. We train 5 epochs for each taskwith the learning rate 5e-4.
- **FT**: We fine-tune the whole model, including the encoder and classifier. We train 3 epochs for each task in BERT, with the learning rate 2e-5.
- **Adapter**: we select learning rates from {5e-5, 1e-4, 1e-3} and train {5, 20} epochs for each task. For all continual learning tasks, we train with the learning rate 5e-5 and 20 epochs. The rank of Adapter projection r is 32 as suggested in the original paper.
- **LoRA (C-LoRA)**: we select learning rates from {5e-4, 1e-3} and train LoRA for {5, 8} epochs for each task.
- **Wire-Fixed and Wire-Neigh**: we select learning rates from {2e-4, 5e-4, 1e-3} and train {5, 8} epochs for each task. The rank of the learnable key matrix is 8. For Wire-Neigh, the number of neighbors is 5. In practice, they are randomly sampled from a larger neighborhood ranging from {10, 20,50,100}. We set the mixing ratio as $s = 0.1$.
- **IDBR**: We train IDBR with the learning rate 3e-5 for 3 epoches per task. We follow the k-means memory selection rule, and the replay batch size is 16 (training batch size) $\times$ number of tasks in the memory.
- **CTR**: We follow the settings in the original paper, training 5 epochs for each task.
- **L2P**: We have the prompt pool with 100 prompt tokens and select 50 of them to prepend to the input. We train the model with the learning rate 1e-3 for 20 epochs for each task.
- **CODA**: We have the prompt component size 20 for each task, and set the prompt length as 20. We train the model with the learning rate 1e-3 for 10 epochs for each task.
- **ER**: We apply sparse experience replay with 1% replay ratio. At each replay time, we sample 32 samples from the memory and perform one-step gradient descent based on them.
- **A-GEM**: We store all previous data in the memory. At each gradient step, we randomly extract 32 samples from the memory and apply the A-GEM gradient projection.
- **MBPA++**: We fine-tune the model with ER and then adapt the model at the inference time. At the inference time, we retrieve 32 nearest samples in the memory for local adaptation.

B Evaluation Metrics

Recall@k Denote the set of task-related tokens of the i-th sample as rel_i, the set of top-k tokens predicted from the learned data representation as $\text{pred}_i@k$. The metric Recall@k calculates the proportion of task-related tokens rel_i that are predicted in $\text{pred}_i@k$, which is defined as:

$$\text{Recall}@k = \mathbb{E}_i\big[\frac{|\text{pred}_i@k \cap \text{rel}_i|}{|\text{rel}_i|}\big].$$

Because each data instance in E-SNLI has 5-10 task-related tokens, we use Recall@20 for evaluation.

Average Accuracy and Forgetting We use the average accuracy and average forgetting similar in [8] to evaluate the performance in CL scenarios. The specific definitions are described below.

- **Average Accuracy (*ACC* $\in$ [0,1])**: Let $a_{i,j}$ be the performance of the model on the test set of task j after the model is trained on task i. The average accuracy after training on all task T is:

$$ACC_T = \frac{1}{T}\sum_{j=1}^{T} a_{T,j}.$$

  In this paper, we select T as the end of CL task sequence.

- **Average Forgetting (*FGT* $\in$ [-1,1])**: Denote $f_{i,j}$ as the forgetting on task j after the model is trained on task i. $f_{i,j}$ is calculated by:

$$f_{i,j} = \max_{l \in \{1,\ldots,i-1\}} a_{l,j} - a_{i,j}.$$

And the forgetting after training on the task T is:

$$FGT_T = \frac{1}{T} \sum_{j=1}^{T-1} f_{T,j}.$$

Our forgetting score divides the number T of all tasks instead of $T-1$. We do this to make the above metrics also indicate models' capacities on single tasks, i.e. single-task capacity $\approx ACC_T + FGT_T$.

C Computation Costs

In this section, we discuss the computation costs of alignment models. All our models use shared parameters across tasks, which do not progressively increase parameters. Since the learnable new matrices in our models are all low-ranked, they require limited usage of additional memory. So we focus on the models' time consumption. For all models, we have the number of input tokens as n.

C-LoRA Our controlled LoRA model has the same time complexity as LoRA.

Wire-Fixed In the Wire-Fixed model, besides forwarding the pre-trained model to computing pre-trained token representations, we need extra computation for Eq. 5. Since we only query for the `[CLS]` token, the extra time complexity is $\mathcal{O}(n)$.

Wire-Neigh In Wire-Neigh, the extra time consumption comes from searching for neighbors and updating the neighbor representations. We find neighbors for each token based on their embeddings at the embedding layer. The time complexity is $\mathcal{O}(Vn + VlogV)$ for computing the cosine similarity and sorting, where V is the size of the token vocabulary. Then the k neighbor representations are updated for each layer, with a complexity of $\mathcal{O}(nk^2)$.

In practice, since the embedding layer is fixed in our model, for each data instance we only need to find their neighbors once and then store the neighbor indices for iterative training (i.e., for several training epochs). The neighbor selection can also be accelerated by reducing the search space of neighbor tokens, for example, only searching neighbors from frequently used tokens instead of the whole vocabulary.

Wire-Neigh also needs extra memory to store neighbor representations and update them. To control the extra consumption of memory, we keep the number of neighbors as $k = 5$ and randomly sample neighbors from top-K similar tokens to control the range of the neighborhood. We leave the improvement of Wire-Neigh's efficiency for future works.

Computation induced by PF Another computation costs come in the probing stage, in which we fix the encoder and only train the classifier. This takes less than 40% training time (including language model forwarding) and 30% GPU memory compared to full fine-tuning.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: This paper studies some key factors of forgetting in continual learning and proposes an approach to mitigate that problem.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We included a limitation section (Section 6).

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [NA]

Justification: This paper is not for theoretically proofing something.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We have included our detailed experimental settings in Appendix A.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [No]

Justification: We are cleaning our code and will release the code soon.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyper-parameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: We included details in Appendix A.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: We report mean and standard deviation from 5 runs.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.

- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We included this information in our Appendix A.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

   Answer: [Yes]

   Justification: We followed it.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [NA]

    Justification: Our work does not have negative societal impacts. For positive impacts, since we study the CL problem training tasks sequentially without revisiting previous tasks or data, we have a positive impact on security. However, this is not our key point so we do not discuss it in the paper.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: No risks.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We used open-source datasets.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification: No new assets introduced.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: No these types of studies.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: NA.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.