POET: Prompt Offset Tuning for Continual Human Action Adaptation

Prachi Garg[1], K J Joseph[3], Vineeth N Balasubramanian[3], Necati Cihan Camgoz[2], Chengde Wan[2], Kenrick Kin[2], Weiguang Si[2], Shugao Ma[2], and Fernando De La Torre[1]

[1] Carnegie Mellon University, USA
[2] Meta Reality Labs
[3] Indian Institute of Technology, Hyderabad

Abstract. As extended reality (XR) is redefining how users interact with computing devices, research in human action recognition is gaining prominence. Typically, models deployed on immersive computing devices are static and limited to their default set of classes. The goal of our research is to provide users and developers with the capability to personalize their experience by adding new action classes to their device models continually. Importantly, a user should be able to add new classes in a low-shot and efficient manner, while this process should not require storing or replaying any of user's sensitive training data. We formalize this problem as privacy-aware few-shot continual action recognition. Towards this end, we propose *POET: Prompt-offset Tuning*. While existing prompt tuning approaches have shown great promise for continual learning of image, text, and video modalities; they demand access to extensively pretrained transformers. Breaking away from this assumption, POET demonstrates the efficacy of prompt tuning a significantly lightweight backbone, pretrained exclusively on the base class data. We propose a novel spatio-temporal learnable prompt offset tuning approach, and are the first to apply such prompt tuning to *Graph Neural Networks*. We contribute two new benchmarks for our new problem setting in human action recognition: (i) NTU RGB+D dataset for activity recognition, and (ii) SHREC-2017 dataset for hand gesture recognition. We find that POET consistently outperforms comprehensive benchmarks. [4]

Keywords: 3D Skeleton Activity Recognition · Extended Reality (XR) · Continual Learning · Prompt Tuning.

1 Introduction

A key input modality to virtual, augmented and mixed reality (often together termed as extended reality, XR) devices today is through recognizing human

[4] Source Code at https://github.com/humansensinglab/POET-continual-action-recognition

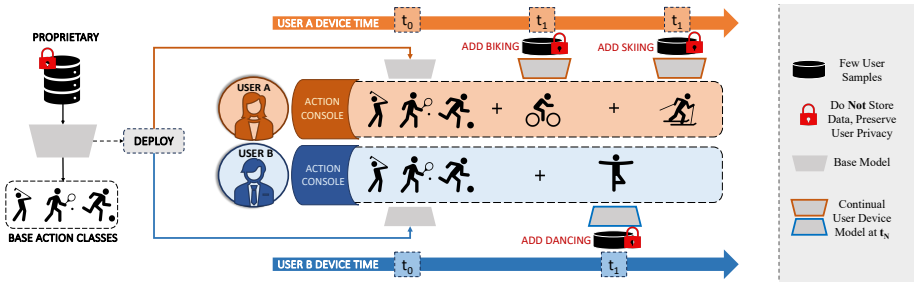


Fig. 1: Proposed POET method **continually adapts** skeleton-based human action recognition models pretrained on a pre-defined set of categories **to new user categories with few training examples**. Users can thus expand the capabilities of XR systems with novel action classes by providing a few examples of each new class. We discard the user-sensitive data as soon as the model is updated on the new categories.

activity and hand gestures based on body and hand pose estimates. Recognizing human actions[5] facilitates seamless user interactions in head-mounted XR devices such as the Meta Quest 3 and Apple Vision Pro. If the provided action recognition models are static, then developers and users are limited to a pre-defined set of action categories. With the growing use of such devices in new contexts and the increasing demand for personalized technology delivery, there is an impending need to enable the action recognition models in such systems to adapt and learn new user actions over time. Defining their own action categories allows users to customize their experience and expand the functionality of their XR devices. Addressing this need is the primary objective of this work.

Adapting human action models to new user categories over time faces a few challenges. Firstly, the model must be capable of learning new actions with minimal amount of training data so users can add new classes by providing just a few training examples per class. Secondly, due to the increasing use of XR devices for personal assistance, there is a need for privacy preservation in user action recognition-based pipelines [2, 14]. Hence, the adaptation of such action recognition models to new user categories must also be 'data-free', i.e., it cannot store and replay previously seen user training data in subsequent continual sessions. Considering these requirements, we leverage the recent success of 'data-free' prompt-based learning [49] and propose a new spatio-temporal prompt offset tuning approach to efficiently adapt the default model without finetuning.

Human action recognition systems are moving to skeleton-based approaches, especially in applications that require low-shot action recognition capabilities such as medical action recognition [27, 56]. Skeletons offer a robust and compact alternative to videos in such low-shot regimes, due to their relatively low dimensionality and lesser variance under background conditions. While there have been a wide variety of efforts in skeleton-based human action recognition over the years [36, 52, 53], there have been fewer efforts on adapting such models to newer user categories. Efforts like [1, 22] attempted to continually learn new user

[5] We use human action as an umbrella term for both hand gesture and body activity in this work for ease of presentation.

categories over time in skeleton-based human action recognition, but relied on fully-supervised data for the new classes. On the other hand, few-shot learning works [27, 46, 56] adapt a pretrained skeleton-based action recognition model to new data, but without explicitly retaining past categories. In this work, we seek to learn new user categories in trained human action models with *very few labeled samples* for the new classes, while being *data-free* (not storing samples from previously trained categories). Fig 1 summarizes our overall objective. One could view our setting as privacy-aware few-shot continual learning for skeleton-based action recognition.

To this end, we propose a prompt offset tuning methodology that can be integrated with existing backbone architectures for skeleton-based human action recognition. Our learnable (soft) prompts are selected from a shared knowledge pool of prompts based on an input instance dependent attention mechanism. In particular, we propose prompt selection using an ordered query-key matching that enables a temporal *prompt* frame order selection consistent with the input instance. We show that such an approach allows us to learn new user categories without having to store data from past classes, without overwriting the pre-existing categories. To the best of our knowledge, this is the first effort on leveraging prompt tuning for skeleton-based models, as well as on spatio-temporal prompt selection and tuning.

Our key contributions are summarized below:

- We formalize a novel problem setting which continually adapts human action models to new user categories over time, in a privacy aware manner.
- To address this problem, we propose a novel spatio-temporal Prompt OffsEt Tuning methodology (POET). In particular, it is designed to seamlessly plug-and-play with a pretrained model's input embedding, without any significant architectural changes.
- Our comprehensive experimental evaluation on two benchmark datasets brings out the efficacy of our proposed approach.

2 Related Works

2.1 Prompt Tuning

The idea of prompting, as it originated from Large Language Models (LLMs), is to include additional information, known as a text prompt, to condition the model's input for generating an output relevant to the prompt. Instead of applying a discrete, pre-defined 'hard' language prompt token, *prompt and prefix tuning* [20, 23] formalized the concept of applying 'soft prompts' to the input. A set of learnable parameters are prepended (concatenated) to the input text and trained along with the classifier while keeping the backbone parameters frozen. Similar to prompt tuning of LLMs, recent works have popularized prompt tuning of ViTs [16] as an effective way of adapting large pretrained models to downstream tasks [49, 57]. However, it remains unexplored and undefined (to the best of our knowledge) for *non-transformer* architectures such as GNNs.

2.2 Prompt Tuning for Continual Learning

Prompt tuning provides a simple and cost-effective way of learning task-specific signal condensed into 'soft prompts'. For continual learning, training a set of prompts for each sequential task provides a natural alternative to storing privacy violating exemplars and replaying them. Training task-specific prompts for each sequential task is straightforward when authors assume access to task identity at both train and inference time, like in Progressive Prompts [34]. However, if task identity is unavailable at inference, the model will not know which task's prompts or classifier to use for evaluating a test sample. In this respect, S-prompts [47] and A-la-carte prompt tuning (APT) [4] learn an independent set of prompts for each domain/task and employ a KNN-based search for domain/task identity at test time. Since these methods learn stand-alone prompts for every task, the prompt feature space is task-specific, and there is no forgetting of old knowledge when learning new tasks (by design). At the same time however, these 'no forgetting' prompts *cannot share knowledge* across tasks.

This leads to another ideology for continual prompt tuning, i.e., treat each prompt unit as being a part of a larger **shared (knowledge) pool** of prompts. Then the desired number of prompt units can be selected from the pool, conditioned on the input instance itself [41, 48, 49]. Given the scarcity of new data in our setting, we hypothesize that sharing of knowledge will benefit new tasks and draw inspiration from this line of works. Most recently, Adaptive Prompt Generator (APG) [42] challenges the intensive ImageNet21K pre-training assumption as it prompts a ViT pretrained only on the continual benchmark's base class data (similar to us). However, they use replay and knowledge distillation-style 'anti-forgetting learning', in addition to using prompts. Even though our backbone is trained only on the base classes, we propose a **simple prompt tuning-only** strategy to counter forgetting. This implies that a prompt strategy is all we need to continually add new action semantics in a few-shot manner.

2.3 Few-Shot Class Incremental Learning

FSCIL is a challenging continual learning setting where a model overfits to new classes, with the simultaneous heightened (often complete) forgetting of old knowledge as soon as the base model is fine-tuned on few-shot data [10, 43]. Since the backbone feature extractor is the only source of previously seen knowledge, if it is updated, knowledge is lost forever. Typically, existing works decouple the learning of (backbone) feature representations from the classifier by *learning* the model *only on the base data* and relying on non-parametric class-mean classifiers for classification in subsequent steps [13, 31, 55]. This leads to a feature-classifier misalignment issue [32, 51] because new class prototypes are extracted from a backbone representation trained only on the base classes. We hypothesize that optimizing input prompt vectors along with a dynamically expanding parametric classifier on top of a frozen backbone can alleviate this misalignment issue. Our work not only provides a fresh perspective into FSCIL, but to our best knowledge is also the only work not designed for and evaluated on image benchmarks.

3 Preliminaries

Skeleton Action Recognition Using Graph Representations. Our input $\mathbf{X} \in \mathbb{R}^{T \times J \times 3}$ is a video sequence of T frames, each frame containing J joints of the human body (25 joints) or hand skeleton (22 joints) in 3D Cartesian coordinate system. Such a skeleton action sequence is naturally represented as a graph topology $G = \{\mathcal{V}, \mathcal{E}\}$ with $\mathcal{V}$ vertices and $\mathcal{E}$ edges. Graphs are modeled using Graph Neural Networks (GNNs) [12], which can either be sparse graph convolutional networks (GCN) or fully connected graph transformers (GT). Our main model (a GNN) is defined as $f(\mathbf{X}) = f_c \circ f_g \circ f_e(\mathbf{X})$ (as also shown in Fig. 2). Input $\mathbf{X}$ is first passed through an input embedding layer f_e to get an embedding of human joints $\mathbf{X_e} = f_e(\mathbf{X}), \mathbf{X_e} \in \mathbb{R}^{T \times J \times C_e}$, with feature dimension C_e. $\mathbf{X_e}$ is further passed to a graph feature extractor f_g composed of a stack of convolutional layers (in GCNs) or attention layers (in GTs), and finally a classifier f_c which predicts the action class label $\mathbf{y}$. In POET, we propose to attach learnable parameters $\mathbf{P_T}$ (called prompt offsets) to the embedding $\mathbf{X_e}$.

Problem Definition. Given a default (pre)trained model deployed on a user's device, we would like to extend this model to new action classes over T subsequent user sessions (also called tasks) $\{\mathcal{US}^{(1)}, ..., \mathcal{US}^{(T)}\}$[6]. In each user session $\mathcal{US}^{(t)}$, the model learns a dataset $\mathcal{D}^{(t)} = (\mathbf{X}_i^t, \mathbf{y}_i^t)_{i=1}^{|\mathcal{D}^{(t)}|}$ of skeleton action sequence and label pairs provided by the user, $\mathbf{X}_i^t \in \mathbb{R}^{T \times J \times 3}$, $y_i^t \in \mathbb{R}^{\mathcal{Y}^{(t)}}$. In each session, the user typically provides a few training instances F (e.g. $F \leq 5$) for each of the N new classes being added, such that $|\mathcal{D}^{(t)}| = NF$. The base (default) model's session $\mathcal{UB}^{(0)}$ is assumed to have a large number of default action classes $\mathcal{Y}^{(0)}$ trained on sufficient data $\mathcal{D}^{(0)}$, which is most often proprietary and cannot be accessed in later user sessions. In each session, the user adds new action classes such that, $\mathcal{Y}^{(t)} \cap \mathcal{Y}^{(t')} = \emptyset, \forall t \neq t'$[7]. Due to the aforementioned privacy constraints, in any training session $\mathcal{US}^{(t)}$, the model has access to only $\mathcal{D}^{(t)}$; after training this data is made inaccessible for use in subsequent sessions (no exemplar or prototypes stored). After training on every new session $\mathcal{US}^{(t)}$, the model is evaluated on the test set of all classes seen so far $\cup_{i=0}^{t} \mathcal{Y}^{(i)}$. The challenge is to alleviate forgetting of old classes while not overfitting to the user-provided new class samples. One could view our setting as privacy-aware few-shot continual action recognition, a problem of practical relevance in human action recognition – which has not received adequate attention.

4 Methodology: Prompt Offset Tuning (POET)

Overview. We propose to prompt tune a base GNN model $f(.)$ by prompts $\mathbf{P_T}$ to address our overall objective. As shown in Fig. 3, for each input instance $\mathbf{X}$, corresponding prompts $\mathbf{P_T}$ are selected from a pool of prompt parameters, using an input-dependent query and key attention mechanism. The selected prompts

[6] User sessions may be spaced at arbitrary time intervals.

[7] We make this assumption considering this is a first of such efforts; allowing for overlapping action classes and users to 'update' older classes would be interesting extensions of our proposed work.

are added to the input feature embedding (and hence the term '*prompt offsets*'), before forwarding to the feature extractor and classifier (shown in Fig. 2).

To this end, our method, POET uses the same number of prompts as the number of temporal frames in the input, to maintain temporal consistency between the prompt and the input. Focusing solely on prompt offsets allows us to adapt the model to subsequent user sessions without having to update the input embeddings or the feature extraction backbone. Our prompt selection mechanism is learnable and trained along with the classifier to make this method simple and efficient.

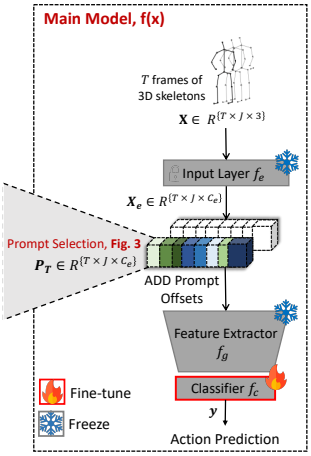


Fig. 2: POET: Prompt-offset Tuning proposes to offset the input feature embedding $\mathbf{X_e}$ of the main model by learnable prompt parameters $\mathbf{P_T}$ for privacy-aware few-shot continual action recognition. We explain prompt selection mechanism in Fig. 3.

What are Prompt Offsets? Learnable (or soft [20]) prompts are parameter vectors in a continuous space which are optimized to adapt the pre-trained frozen backbone f_g to each continual task. We define our spatio-temporal prompt offsets $\mathbf{P_T}$ as a set of T prompts (same in number as skeletal frames in input), each prompt $\boldsymbol{P_i}$ having length equal to the number of joints in a frame J and feature dimension same as input feature embedding $\mathbf{X_e}$, i.e., $\boldsymbol{P_i} \in \mathbb{R}^{J \times C_e}$.

Existing prompt tuning efforts, for example in image classification, focus on concatenating learnable prompts to the input token sequence in transformer architectures [16, 20]. Even though transformers can be generalized to graphs [7,11,29], it is non-trivial to attach prompts to a GNN. This is because transformers can be viewed as treating sentences or images as fully connected graphs where any word (or image patch) can attend to any other word in the sentence [12]. However, our input is a spatio-temporal graph skeleton of the human joint-bone structure with its own edge connectivity. Concatenating prompts along spatial or temporal dimensions would affect the graph semantics, and also affect standard training strategies such as a forward pass or backpropagation (especially in GCNs). Hence, we attach the selected prompts $\mathbf{P_T}$ to the corresponding input feature embedding $\mathbf{X_e}$ via a prompt attachment operator $f_p(.)$. The class logit distribution $\mathbf{y}$ is thus obtained as:

$$\mathbf{y} = f(\mathbf{X}, \mathbf{P_T}) = f_c \circ f_g \circ f_p(f_e(\mathbf{X}), \mathbf{P_T}) \tag{1}$$

In every user session $t > 0$, the classifier output dimension expands by N to accommodate the new action classes. Unlike most existing continual prompt tuning works, our feature extractor backbone f_g is trained only on the base class data $\mathcal{D}^{(0)}$ and is never fine-tuned on classes from new user sessions $\mathcal{US}^{(t)}, t > 0$. After the base session training, parameters of f_g, f_e are frozen.

Prompt Pool Design. As stated in Sec. 2.2, to encourage knowledge sharing across user sessions, we choose to construct a single prompt pool $\mathbf{P}$ which encodes

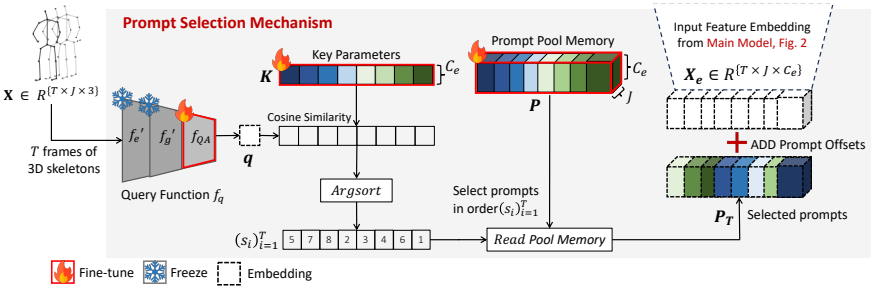


Fig. 3: Selection of our prompts $\mathbf{P_T}$: Input-dependent query $\boldsymbol{q}$ is matched with keys $\boldsymbol{K}$ using *sorted* cosine similarity to get an ordered index sequence $(s_i)_{i=1}^T$ of the top T keys. This ordered index sequence is used to select the corresponding ordered prompt sequence $\mathbf{P_T}$ from prompt pool $\mathbf{P}$. We *add* $\mathbf{P_T}$ to $\mathbf{X_e}$, thereby adding an offset to it. Our experimental evaluation confirms that such an additive spatio-temporal prompt offset can balance the plasticity to learn new classes from a few action samples, while maintaining stability on previously learned classes.

knowledge across the sessions:

$$\mathbf{P} = \{\boldsymbol{P_1}, ..\boldsymbol{P_i}, ..., \boldsymbol{P_M}\}, \qquad \boldsymbol{P_i} \in \mathbb{R}^{J \times C_e}; M = \#\text{prompts at time t} \tag{2}$$

For selecting prompts from this pool (Fig. 3), we construct a bijective key-value codebook, treating prompts in the pool $\mathbf{P}$ as values and defining learnable key vectors $\boldsymbol{K} = \{\boldsymbol{k_1}, .., \boldsymbol{k_i}, .., \boldsymbol{k_M}\}, \boldsymbol{k_i} \in \mathbb{R}^{C_e}$. A cosine similarity matching $\gamma(.)$ between the query $\boldsymbol{q}$ and keys $\boldsymbol{K}$ is used to find indices of the T closest keys $\mathbb{Z}$, which in turn are used to select prompts from the pool:

$$\mathbb{Z} = \underset{T}{\text{argmax}}\, \gamma(f_q(\mathbf{X}), \boldsymbol{K}) \tag{3}$$

This quantization process is enabled by a query function $f_q(.)$, which is a pre-trained encoder that maps an input instance $\mathbf{X}$ to a query $\boldsymbol{q}$ as:

$$\boldsymbol{q} = f_q(\mathbf{X}) = f_{QA} \circ f'_g \circ f'_e(\mathbf{X}), \qquad f_q \colon \mathbb{R}^{T \times J \times 3} \longrightarrow \mathbb{R}^{C_e} \tag{4}$$

where the *query adaptor* f_{QA} is a fully connected layer mapping the f'_g output dimension to the desired prompt embedding dimension C_e.

Coupled Optimization in User Sessions $t > 0$**.** Typically, the argmax operator in Eq. 3 decouples the optimization of keys from the prompt pool and main model as it prevents backpropagation of gradients to the keys (also seen in earlier works such as [48, 49]). However, this approach does not work for our setting, even more so since we assume no large-scale pre-training of our base model. Due to the lack of off-the-shelf availability of large-scale models for skeletal action data, our query function f_q is pretrained only on base class data $\mathcal{D}^{(0)}$. Hence, it becomes important that f_q is updated as the model learns new classes. As shown in red boxes in Figs. 2 and 3, we propose to couple this optimization process such that the overall cross-entropy loss for new tasks updates: (i) the classifier f_c, (ii) selected prompts in $\mathbf{P}$, (iii) selected keys in $\boldsymbol{K}$, as well as (iv) query adaptor f_{QA}. We achieve this by approximating the gradient for $\boldsymbol{K}$ and f_{QA} by the straight-through estimator reparameterization trick as in [3, 44]. We freeze the query feature extractor layers f'_g, f'_e in $t > 0$ to prevent catastrophic forgetting of base knowledge in f_q. Our cross-entropy loss is hence given by:

$$\min_{\theta_{f_{QA}},\theta_{\boldsymbol{K}},\theta_{\mathbf{P}},\theta_{f_c}} \mathcal{L}(f(\mathbf{X},\mathbf{P_T}),\mathbf{y}) \tag{5}$$

To move queries closer to their aligned T keys during training, we use a vector quantization clustering loss inspired from VQ-VAE [44] as:

$$\max_{\theta_{f_{QA}},\theta_{\boldsymbol{K}}} \lambda \sum_{i\in\mathbb{Z}} \gamma(f_q(\mathbf{X}),\boldsymbol{K_i}) \tag{6}$$

where λ is the clustering loss coefficient. Our end-to-end optimization thus establishes a prompt optimization framework which is amenable to prompt tuning when extensive pre-training is not possible. This sets the foundation for our spatio-temporal prompt selection module, described next.

Spatio-Temporal Prompt Selection. In order to ensure that our learned prompts respect temporal information in the input video sequence, we choose the number of selected prompts to be equal to the number of frames in the input video T. After coupling the prompt pool and keys, we observed in our initial experiments with pool size $M > T$ that the same set of prompts get selected across training iterations and user sessions (Fig. 4A). More concretely, as the vector quantization loss (Eqn 6) brings the query close to the selected keys, the same set of active prompts get selected and optimized in each iteration, not using other prompts at all. This is similar to the well-known issue of '*codebook collapse*' in VQ-VAE [9, 50, 54]. Based on this observation, we design two prompt pool update mechanisms in user sessions $t > 0$ as below:

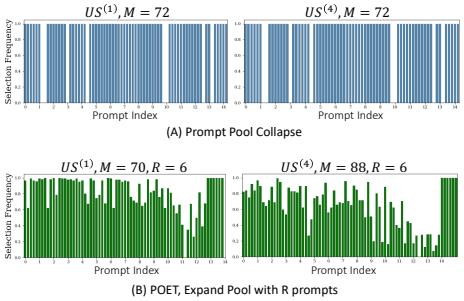


Fig. 4: $M > T$ **Case: Prompt Pool Collapse.** (Top) Certain prompt indices remain unused across user sessions. (Bottom) Our POET pool expansion strategy alleviates pool collapse.

1. **Case 1,** $M = T \forall t$**:** *No pool expansion, Algorithm 1.* All prompts are selected in all tasks. But the *order of their selection* $(s_i)_{i=1}^{T}$ varies with each input instance as we replace Eq. 3 by sorting the cosine similarity before selecting the top T indices as follows:

$$\mathbb{Z} = \underset{(s_i)_{i=1}^{T}}{\text{argsort}}\, \gamma(f_q(\mathbf{X}),\boldsymbol{K}) \tag{7}$$

   In Fig. 5, we visualize the positions occupied by indices in this (sorted) *ordered key index sequence* $(s_i)_{i=1}^{T}$. Entropy increase across tasks $t = 1$ to $t = 4$ (bottom row of figure) shows that our selection mechanism learns to select a unique temporal code for all inputs.
2. **Case 2,** $M = T + (R * t), t > 0$**.** *Expand pool with R prompts.* We also propose an order-aware prompt pool expansion strategy (Appendix B) that selects prompts from an expanded pool in a temporally coherent manner, for $t > 0$. This alleviates prompt pool collapse as shown in Fig. 4B.

Prompt Offset Attachment. Since concatenation is not meaningful for graph data, we use addition as our choice for the prompt attachment operator as:

$$f_p(\mathbf{X_e}, \mathbf{P_T}) = \mathbf{X_e} + \mathbf{P_T} \tag{8}$$

Hence, we call our approach as *prompt offset tuning*. We also study this empirically through experiments that support this choice in Sec. 6.

Interpreting Prompt Offset Tuning of GNNs. Our additive prompt offsets are open to interpretation, as shown in Fig. 5. (i) Adding our selected prompts $\mathbf{P_T}$ to input feature embedding $\mathbf{X_e}$ acts like an input-dependent transformation for spatio-temporal joints. (ii) As our prompts have same size as $\mathbf{X_e}$, it can also be thought of as a learned prompt encoding, bearing similarity with learnable position encoding works [12, 24, 26]. Our purpose is different however as prompt offsets seek to dynamically condition the input for adapting the backbone continually, instead of learning positions. (iii) POET also bears similarity with auto-decoders like DeepSDF [30] which learn latent codes for each style or shape and use relevant codes along with a frozen decoder at inference. (iv) Prompt tuning can also be thought of as a *parameter isolation* technique for continual learning [28, 34, 35, 38]. POET's ordered prompt selection as seen in Fig. 5 learns to *isolate* the relevant sequence of prompts for each input action sequence.

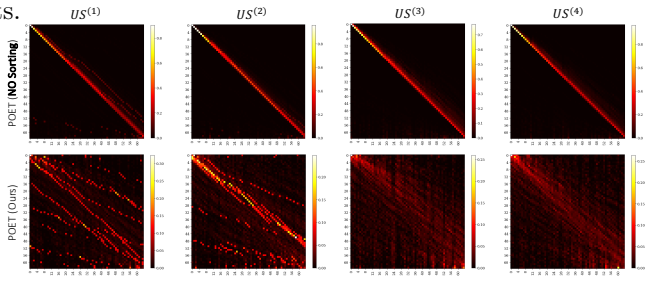


Fig. 5: Here we visualize the order $(s_i)_{i=1}^T$ in which the $M = 64$ prompts in the pool are selected at train time, across 4 user sessions $\mathcal{US}^{(t)}$. X-axis: prompt index, Y-axis: index position in selected sequence. *Top:* The no sorting case uses the default sequence (hence diagonal matrices), giving equal importance to all prompts. *Bottom (Our Method):* Even though the same 64 prompts are selected and updated, the ordering is temporally unique and consistent with input.

Algorithm 1 POET at Train Time, $t > 0$ (Case 1 $M = T$, No pool expansion)

Input: Query function f_q, keys $\boldsymbol{K} = \{\boldsymbol{k_j}\}_{j=1}^T$, prompt pool $\mathbf{P} = \{\boldsymbol{P_j}\}_{j=1}^T$; main model f_e, f_g, f_c
Initialize: $\mathbf{P}$, $\boldsymbol{K}$ from $t-1$; Expand f_c by N new classes. Initialize f_c as: (i) copy f_c^{old} weights, (ii) $f_c^{new} \leftarrow Mean(f_c^{old})$
Freeze: query layers f_g', f_e'; main model layers f_e, f_g
for epochs and batch $(\mathbf{X}_i^t, \mathbf{y}_i^t)_{i=1}^{NK}$ do
1. Get query feature $\boldsymbol{q}$ (Eq. 4) ; Compute $\gamma(.)$ b/w query $\boldsymbol{q}$ and keys $\boldsymbol{K}$
2. Sort $\gamma(.)$; Get ordered key index sequence $(s_i)_{i=1}^T$ (Eq. 7)
3. Read pool memory $\mathbf{P}$ in order $(s_i)_{i=1}^T \rightarrow$ Get prompt offsets $\mathbf{P_T}$
4. Get $\mathbf{X_e}$; **Add** $\mathbf{P_T}$ to it (Eq. 8); get prediction $\mathbf{y}$ from prompted input (Eq. 1)
5. Use cross entropy loss (Equation 5) to **update** $f_{QA}, \boldsymbol{K}, \mathbf{P}, f_c$
6. Use clustering loss (Equation 6) to **update** f_{QA} and $\boldsymbol{K}$

end // See $t = 0$ training protocol in Algorithm 2 in Appendix

5 Experiments and Results

Datasets. We evaluated our method on well-known action recognition datasets[8]: (i) activity recognition on the NTU RGB+D dataset [39]; and (ii) hand gesture

[8] The datasets used in this work were accessed and processed at and by CMU. They were not accessed, processed, stored, or maintained at Meta.

recognition on the SHREC-2017 dataset [40]. As we introduce a new problem setting in human action recognition, we contribute two new benchmarks to the community for this setting, on the NTU RGB+D and SHREC-2017 datasets.

For the NTU RGB+D dataset, we divide the 60 daily action categories into 40 base classes, learning the remaining 20 classes in subsequent user sessions. In few-shot learning parlance, our protocol is 4-task 5-way 5-shot, i.e. 5 novel classes using 5 user training instances in 4 user sessions. Each input 3D skeleton sequence has 64 temporal frames, each consisting of 25 body keypoints, such that $x \in \mathcal{R}^{64\times25\times3}$. We use the spatio-temporal GCN, CTR-GCN [8], as the architecture for NTU RGB+D, where we choose the joint input modality for better interpretability of prompt tuning.

For SHREC-2017, we divide the 14 fine-grained hand gesture classes into 8 base classes and 6 classes learned in subsequent user sessions. This is done in a 3-task 2-way 5-shot protocol, i.e. 2 novel classes using 5 user training instances in 3 user sessions. For each input instance of SHREC-2017, we use 8 temporal frames each having 22 hand keypoints, such that input $x \in \mathcal{R}^{8\times22\times3}$. We use a fully-connected graph transformer backbone, DG-STA [7] for SHREC-2017. We select DG-STA due to easily reproducible code and to validate if our method POET works equally well across graph convolutional networks and graph transformers.

Evaluation Metrics. Following earlier work in similar settings [31], we report: (i) Average accuracy '*Avg*' of all classes seen so far, and (ii) Harmonic Mean A_{HM} between '*accuracy only on Old classes*' and '*accuracy only on New classes*' after learning each new user session. Note that the average accuracy tends to be biased towards the base session $\mathcal{T}^{(0)}$ performance due to more number of base classes. A higher A_{HM} implies better stability-plasticity trade-off between new task performance and old tasks' retention. Unlike many earlier CIL efforts, we report accuracy for both *Old* and *New* classes in each user session for transparency.

Implementation Details. We observe that a key source of forgetting in our setting is from the classifier as the logits tend to become heavily biased towards the few-shot samples of new classes. We use a cosine classifier for activity recognition experiments on CTR-GCN. For gesture recognition on the lightweight DG-STA, we use a standard fully-connected layer as classifier, but freeze old class parameters in the classifier by zeroing their gradients. We attach prompts after the 1st layer of DG-STA and 1st CTR-GC block of CTR-GCN. For both datasets, we have equal or higher learning rates in user sessions when compared to the base model's training in order to accommodate new knowledge in the model (for better plasticity). For exact implementation details (including learning rates, epochs, hyperparameter analysis, and backward forgetting metric), see Appendix A. In earlier efforts that more generally tune prompts for class-incremental learning [41, 45, 47–49], it is common to rely on an ImageNet21K pretrained ViT [37] or CLIP [33] as the backbone. However, such backbones do not exist for skeleton-based human action recognition. Our base feature extractor is hence trained on the base session dataset itself without *any* pretraining, making this one of the first efforts of prompt tuning without extensive pretraining (scale of data 3-5 times lower order of magnitude).

Results. Since there are no existing baselines for our proposed setting in skeletal action recognition, we compare our method by adapting continual learning (CL) baselines to skeletal data in Sec 5.1, Tables 1, 2. We first compare POET with prompt tuning based class-incremental learning (CIL) approaches originally designed for images (L2P [49], CODA-P [41], APT [4]) and find that it has very low performance on new classes as they do not update their query function. We find any fine-tuning or knowledge distillation based approaches (LWF [25], EWC [17], LUCIR [15]) lead to rapid forgetting of base knowledge as the model overfits to user's few-shots. We also compare with multiple variants of Feature Extraction (FE) to check if prompts truly have merit (POET=FE+Prompts) and provide upper bound baselines. In Sec 6, we first show the importance of prompts in POET by removing the prompts. We discuss the value of our coupled optimization, query function update and *ordered key index selection* in our prompt selection ablation Tab 3. We also study the impact of proposed additive prompt tuning as compared to other possible prompt attachments f_p in Tab 4.

5.1 Comparison with State-of-the-Art

POET sets the SOTA on existing prompt tuning works (Tab 1,2). We adapt three standard CIL works that prompt tune ViTs for images - L2P [49], CODA-P [41] and APT [4] to our setting. L2P and CODA-P share prompt pool across tasks (similar to us), whereas APT learns task-specific prompts. L2P decouples the optimization of keys from the prompt pool and concatenates the selected prompts. Since concatenation is not defined for our GNN backbone, we adapt these SOTA to our setting by concatenating along the temporal dimension (**L2P*, CODA-P***). CODA-P [41] couples keys with the prompt pool by using a cosine similarity weighing over all prompts in the pool, forming a 'soft prompt selection', different from our 'ordered hard prompt selection'. In **APT**, we train prompt-classifier pairs for each continual task separately ($\hat{}$ denotes task-specific), and use task identity at test time. See details in Appendix A. These methods by design rely on extensively pretrained (ImageNet21k) query functions which does not require updates; and require full supervision on new classes, perhaps explaining their poor 'New' accuracy in our few-shot setting.
Standard Continual learning Baselines. We compared with two well established knowledge-distillation approaches, learning without forgetting (**LWF**) and **LUCIR**. Both of them perform poorly on both old and new classes. **EWC** [17] learns better on new but does not retain old knowledge. We conclude that any CL method that fine-tunes the backbone feature representation in subsequent sessions $t > 0$ will not be able to retain base/old class knowledge (a finding consistent with existing FSCIL literature for images [10,43]). We also adapt and compare with one of the latest FSCIL baselines **ALICE** [31], originally developed for image classification benchmarks on our gesture recognition benchmark in Table 2. Note the high retention of base task performance (due to non-parametric classifier on top of frozen base model). However, it suffers from poor plasticity and adaptation to new classes. This is the issue of feature-classifier misalignment that we hoped to alleviate through prompt tuning.

Table 1: Activity Recognition Results (%, ↑), Comparison with SOTA: NTU RGB+D [39] dataset on CTR-GCN [8] backbone. After training on each incremental task, we report Average of all classes seen so far ('Avg'). We also report (i) A_{HM}, (ii) old classes accuracy ('Old'), (iii) new classes accuracy ('New') in the last session. We report Mean and STD across 10 sets of 5-shots. *POET achieves the best stability-plasticity trade-off across all baselines indicated by the* $A_{HM} = 56.3\%$. *POET also has the highest Avg across all user sessions outside of upper bound baselines in orange.*

	$\mathcal{UB}^{(0)}$	$\mathcal{US}^{(1)}$	$\mathcal{US}^{(2)}$	$\mathcal{US}^{(3)}$	$\mathcal{US}^{(4)}$			
Method	Base (↑)	Avg (↑)	Avg (↑)	Avg (↑)	Old (↑)	New (↑)	Avg (↑)	A_{HM} (↑)
Upper Bounds								
Joint (Oracle)	88.4	79.0	71.0	66.8			**63.5**	
Joint POET (Oracle)							**67.2**	
FE, Task-Specific^	88.4	70.1 ± 2.6	52.5 ± 5.8	44.8 ± 5.0	70.3 ± 2.1	46.7 ± 2.0	NA	NA
FE+Replay	88.4	82.4 ± 1.1	78.2 ± 1.2	74.5 ± 1.2	**73.1 ± 1.0**	43.3 ± 3.3	**70.6 ± 1.2**	54.3 ± 2.6
Continual Linear Probing								
FE	88.4	72.0 ± 1.1	60.4 ± 2.4	47.7 ± 2.1	40.0 ± 1.6	51.0 ± 2.3	40.9 ± 1.4	44.8 ± 1.1
FE, Frozen	88.4	76.1 ± 1.0	52.4 ± 4.1	38.3 ± 2.7	28.4 ± 1.6	22.4 ± 4.5	27.9 ± 1.4	24.8 ± 3.0
FE+Replay†	88.4	72.0 ± 1.5	59.5 ± 4.0	58.7 ± 2.8	56.7 ± 2.5	34.7 ± 5.6	54.9 ± 2.7	42.8 ± 4.4
FT	88.4	6.2 ± 1.4	4.3 ± 1.5	2.8 ± 1.0	0.2 ± 0.5	36.0 ± 10.1	3.2 ± 0.8	0.3 ± 1.0
Standard Continual Learning								
LWF [25]	88.4	6.2 ± 1.5	2.8 ± 0.7	3.7 ± 1.3	0.0 ± 0.0	38.9 ± 8.8	3.2 ± 0.7	0.0 ± 0.0
EWC [17]	88.4	6.6 ± 1.5	4.1 ± 1.4	3.1 ± 0.9	0.0 ± 0.0	42.1 ± 9.5	3.5 ± 0.8	0.0 ± 0.0
Experience Replay	88.4	35.1 ± 8.3	50.6 ± 5.0	60.6 ± 5.4	54.6 ± 6.5	43.7 ± 14.6	53.7 ± 7.1	47.8 ± 11.2
Experience Replay†	88.4	6.2 ± 1.5	9.0 ± 2.6	11.2 ± 3.0	10.9 ± 2.6	34.6 ± 7.9	12.9 ± 3.0	16.3 ± 3.5
LUCIR [15]	87.9	4.3 ± 2.1	4.1 ± 1.3	2.7 ± 0.8	0.2 ± 0.4	26.0 ± 9.2	2.3 ± 0.9	0.4 ± 0.8
Continual Prompt Tuning								
CODA-P [41]*	87.4	76.1 ± 1.0	66.7 ± 1.3	58.6 ± 2.7	56.5 ± 2.9	0.5 ± 0.4	51.8 ± 2.7	1.1 ± 0.7
L2P [49]*	88.6	78.9 ± 0.1	71.0 ± 1.0	64.2 ± 0.1	62.0 ± 0.7	0.0 ± 0.0	56.8 ± 0.6	0.0 ± 0.0
APT [4]^	86.6	27.3 ± 1.6	30.8 ± 3.4	37.6 ± 2.3	NA	33.4 ± 2.0	NA	NA
POET (Ours)	87.9	**82.3 ± 0.6**	**76.8 ± 0.9**	**68.4 ± 0.7**	**57.2 ± 1.0**	**55.8 ± 5.9**	**57.1 ± 1.1**	**56.3 ± 3.2**

Fine-tuning (FE) and Feature Extraction (FE) Baselines. We implement standard continual learning baselines to understand stability-plasticity trade-offs in our new benchmarks. In all these baselines, we expand the classifier output dimension by N new classes. In **'FT (Fine-Tuning)'**, we tune all model parameters on cross entropy loss of new task. FSCIL is challenging for this modality as old task performance sharply reduces to zero starting from $\mathcal{US}^{(1)}$ as model overfits to user's few-shots. **'FE (Feature Extraction)'**[9] differs from FT as we freeze the feature extractor to preserve base knowledge. This serves as a competitive baseline in our findings. In **'FE, frozen'**, we zero out the gradients of previous class weights in classifier f_c to prevent forgetting from the classifier. 'FE' and 'FE, Frozen' exhibit different New-Old trade-offs in Tables 1, 2 because the scale of pretraining is different (gesture more lightweight than activity).

Upper-bound baselines, top section Tables 1, 2. In **'Joint (oracle)'** experiment, we train on all task data at the same time in a multi-task (non-sequential) manner. Training POET in a multi-task manner (**'Joint POET'**) outperforms **'Joint Oracle'** demonstrating the strength of our approach. In addition to these *generalist* upper bounds, we point out that '**FE, Task-specific^**' is a competitive *specialist* upper bound. In this, we perform feature extraction from base model to each task individually, storing separate task-specific models ($\mathcal{US}^{(0)} \rightarrow \mathcal{US}^{(i)}$, $i > 0$). POET outperforms 'New' accuracy compared with this baseline, achieving a forward transfer on each $t > 0$. This indicates that prompt

[9] 'FE' is the same as 'w/o prompts' in Table 3. We highlight key baselines in gray color.

Table 2: Gesture Recognition Results (%, ↑), Comparison with SOTA: SHREC 2017 [40] dataset on DG-STA [7] graph transformer backbone. Reporting mean and standard deviation across 5 runs. **POET achieves best** $A_{HM} = 56.2\%$.

	$\mathcal{UB}^{(0)}$	$\mathcal{US}^{(1)}$	$\mathcal{US}^{(2)}$	$\mathcal{US}^{(3)}$			
Method	Base (↑)	Avg (↑)	Avg (↑)	Old (↑)	New (↑)	Avg (↑)	A_{HM} (↑)
Joint (Oracle)	88.8	79.4 ± 0.7	77.3 ± 2.1			**70.9 ± 1.2**	**62.4 ± 0.4**
FT	88.8	20.3 ± 0.8	12.4 ± 2.1	0.0 ± 0.0	85.8 ± 9.4	13.4 ± 1.5	0.0 ± 0.0
FE	88.8	62.7 ± 2.4	41.9 ± 6.9	17.5 ± 5.1	77.3 ± 8.8	26.8 ± 3.4	28.5 ± 6.4
FE, Frozen	88.8	71.3 ± 1.9	61.4 ± 2.7	44.7 ± 3.2	54.5 ± 6.7	46.2 ± 2.7	49.1 ± 4.3
LWF [25]	88.8	20.2 ± 1.4	12.5 ± 1.0	0.0 ± 0.0	**88.4 ± 13.7**	13.8 ± 2.1	0.0 ± 0.0
L2P [49]**	88.8	20.3 ± 5.9	10.5 ± 4.8	8.2 ± 4.0	6.9 ± 8.5	7.9 ± 3.9	7.5 ± 5.5
CODA-P [41]**	87.7	15.6 ± 4.5	11.6 ± 1.9	7.9 ± 1.8	14.1 ± 21.4	8.8 ± 2.4	10.1 ± 3.2
ALICE [31]	92.1	72.4 ± 5.7	63.3 ± 7.6	**62.5 ± 6.8**	11.9 ± 9.9	**54.6 ± 6.9**	20.0 ± 8.1
POET (Ours)	91.9	73.2 ± 3.7	61.9 ± 1.8	45.9 ± 2.6	72.4 ± 7.1	50.0 ± 1.6	**56.2 ± 1.6**

tuning benefits New performance due to the pre-existing knowledge in the shared knowledge pool. Avg in sessions $0 < t < 4$ indicates New for task-specific^ .

Experience Replay Baselines, Tab 1. Even though our privacy-aware setting prohibits previous data replay, we compare with **'Experience Replay'** (store and replay 5-samples of base and incremental sessions) and **'Experience Replay†'** (replay only previous incremental sessions) for completeness. **'FE+Replay'** serves as the best upper bound (even better than Experience Replay as we are freezing backbone in addition to replay). It is noteworthy that POET (which is FE+prompts) learns an implicit 'data-free' form of prompt pool memory, and yet has a better A_{HM} trade-off as compared to explicitly stored and replayed samples from previous classes in FE+replay.

6 Ablation Studies and Analysis

Importance of prompts in POET. First, we *consider the contribution of prompt offsets in POET*. Since we only attach prompts to address continual learning in POET, removing prompts gives the Feature Extraction (FE) baseline ('w/o prompts', Table 3) where the backbone is frozen after base training and only the classifier is expanded and updated on classification loss of new classes. POET improves both, 'Old' (↑ 20.1%) and 'New' (↑ 10.6%) marked in blue.

Prompt Selection Mechanism. In Table 3, we investigate our prompt selection mechanism and optimization choices. The **'w/o coupled optim.'** experiment is a direct comparison of our additive *prompt attachment* with the de-coupled optimization in L2P [49]. Updating key parameters but keeping only query adaptor QA frozen after $\mathcal{UB}^{(0)}$ training (**'w/o QA update'**) reduces 'New' only performance of $\mathcal{US}^{(4)}$ by 4.6% as the query function stays fixed at base session learning and is not discriminative towards new classes. **'W/o clustering loss'** from Eq. 6, performance drops starting from $\mathcal{UB}^{(0)}$ itself. The only difference

Table 3: Prompt Selection Mechanism Analysis on NTU RGB+D dataset (%, ↑): 'w/o' denotes removing that component from POET, numbers in brackets are wrt *POET* ($M = T$) experiment. 'Avg' accuracy is biased towards 'Old' classes accuracy, A_{HM} is good indicator of trade-off between 'New' and 'Old'.

NTU RGB+D	$\mathcal{UB}^{(0)}$	$\mathcal{US}^{(1)}$	$\mathcal{US}^{(2)}$	$\mathcal{US}^{(3)}$	$\mathcal{US}^{(4)}$			
Method	Base	Avg	Avg	Avg	Old	New	Avg	A_{HM}
w/o prompts	88.4	74.5	66.3	49.5	39.2 (-20.1)	46.8 (-10.6)	39.9	42.7
w/o coupled optim.	88.0	82.8	75.3	65.8	56.5 (-2.8)	51.3 (-6.1)	56.1	53.8
w/o clustering loss	85.5	81.6	74.3	64.5	**62.0** (+2.7)	18.2 (-39.2)	57.0	28.1
w/o QA update	87.9	82.8	**77.4**	**69.1**	59.4 (+0.1)	52.8 (-4.6)	58.7	55.9
w/o sorting	88.2	82.2	75.2	68.8	59.9 (+0.6)	46.6 (-10.8)	58.8	52.4
POET ($M > T$)	87.9	82.7	77.2	68.8	60.3 (+1.0)	54.4 (-3.0)	**59.8**	57.2
POET ($M = T$)	87.9	**82.8**	76.8	68.6	59.3	**57.4**	**59.2**	**58.3**

between the experiment **'w/o sorting'** and 'POET (M=T)' is that we do not *sort* the cosine similarity before selecting top T indices (same as Fig 5). The 10.8% ↑ in 'New' performance validates that our prompt selection mechanism is learning to chose a distinct temporal ordering for prompt tuning of new input samples. With pool expansion (**'POET,** $M > T$**'**), we get more flexibility in the stability-plasticity trade-offs depending on how many new prompts we attach. For $R = 6$, 'Old' is improved. In Table 3, we keep POET's additive prompt attachment and only vary prompt selection.

Prompt Attachment Mechanism. In Table 4, we keep our end-to-end optimization and ordered prompt selection as a constant and ablate prompt shape and attachment operator $f_p(.)$. Drawing a parallel with transformers which concatenate prompts along the token dimension, we conduct experiments concatenating prompts along the (i) temporal dimension of the skeleton input feature embedding $\mathbf{X_e}$ (*'CONCAT temporal'*) and (ii) feature dimension C_e (*'CONCAT feature'*). We find that addition works better than concatenation and cross attention. We also verify our hypothesis that selecting the same number of prompts as the input temporal dimension ($T = 64$ for NTU RGB+D and $T = 8$ for SHREC-2017) yields better results as compared to adding the same prompt frame to each input embedding frame (*'Addition* $T' = 1$*'*).

Table 4: Prompt Attachment Analysis (%, ↑): The best prompt attachment choice $f_p(.)$ is *Adding* #prompts same as #input frames (T=64).

NTU RGB+D	$\mathcal{US}^{(0)}$	$\mathcal{US}^{(1)}$	$\mathcal{US}^{(2)}$	$\mathcal{US}^{(3)}$	$\mathcal{US}^{(4)}$			
Method	Base	Avg	Avg	Avg	Old	New	Avg	A_{HM}
CONCAT temporal, $T' = 64$	88.6	70.3	62.4	49.8	33.6	50.5	35.1	40.3
CONCAT feature, $T' = 64$	87.7	82.4	75.5	66.9	57.1	41.5	56.0	48.1
Cross Attention, $T' = 64$	82.9	77.4	72.2	65.0	57.1	32.3	55.0	41.2
ADD, $T' = 1$	**88.7**	73.3	62.7	45.5	33.7	47.0	34.8	39.3
ADD, $T' = 64$ (Ours)	87.9	**82.8**	**76.8**	**68.6**	59.3	**57.4**	59.2	**58.3**

7 Conclusions and Future Work

The problem of continually adapting human action models to new user categories over time has gained prominence with the rising availability of XR devices. However, this setting poses unique challenges: (i) the user may be able to provide only a few samples for training, and (ii) accessing data from earlier sessions may violate privacy considerations. We hence propose a method based on prompt offset tuning to address this problem in this work. Prompt tuning to address learning over newer tasks has been attempted in recent years. However, these works have: (1) typically been designed for image-based tasks, (2) relied on strongly pretrained transformer backbones, (3) required full supervision for new tasks, and (4) exclusively applied prompt tuning to transformer architectures. This work departs from these four characteristics. Our work demonstrates that prompt offset tuning is a promising option to evolve and adapt skeleton-based human action models to new user classes. The careful design of each component of the proposed methodology finds validation in the promising results across well-known skeleton-based action recognition benchmarks. Our ablation studies and analysis corroborate our design choices in our implementation. Looking ahead, it will be interesting to explore how our approach and its design choices adapt when a "generalist backbone" trained on a large corpus of action recognition data becomes accessible. Extending our method for differential privacy is another interesting direction of future work.

References

1. Aich, S., Ruiz-Santaquiteria, J., Lu, Z., Garg, P., Joseph, K.J., Fernandez, A., Balasubramanian, V.N., Kin, K., Wan, C., Camgoz, N.C., Ma, S., De la Torre, F.: Data-free class-incremental hand gesture recognition. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (2023)
2. Albrecht, J.P.: How the gdpr will change the world. Eur. Data Prot. L. Rev. **2**, 287 (2016)
3. Bengio, Y., Léonard, N., Courville, A.: Estimating or propagating gradients through stochastic neurons for conditional computation. arXiv preprint arXiv:1308.3432 (2013)
4. Bowman, B., Achille, A., Zancato, L., Trager, M., Perera, P., Paolini, G., Soatto, S.: a-la-carte prompt tuning (apt): Combining distinct data via composable prompting. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 14984–14993 (2023)
5. Chaudhry, A., Dokania, P.K., Ajanthan, T., Torr, P.H.: Riemannian walk for incremental learning: Understanding forgetting and intransigence. In: Proceedings of the European conference on computer vision (ECCV). pp. 532–547 (2018)
6. Chaudhry, A., Ranzato, M., Rohrbach, M., Elhoseiny, M.: Efficient lifelong learning with a-gem. arXiv preprint arXiv:1812.00420 (2018)
7. Chen, Y., Zhao, L., Peng, X., Yuan, J., Metaxas, D.N.: Construct dynamic graphs for hand gesture recognition via spatial-temporal attention. In: Proceedings of the British Machine Vision Conference (BMVC) (2019)
8. Chen, Y., Zhang, Z., Yuan, C., Li, B., Deng, Y., Hu, W.: Channel-wise topology refinement graph convolution for skeleton-based action recognition. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 13359–13368 (2021)
9. Dhariwal, P., Jun, H., Payne, C., Kim, J.W., Radford, A., Sutskever, I.: Jukebox: A generative model for music. arXiv preprint arXiv:2005.00341 (2020)
10. Dong, S., Hong, X., Tao, X., Chang, X., Wei, X., Gong, Y.: Few-shot class-incremental learning via relation knowledge distillation. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 35, pp. 1255–1263 (2021)
11. Dwivedi, V.P., Bresson, X.: A generalization of transformer networks to graphs. arXiv preprint arXiv:2012.09699 (2020)
12. Dwivedi, V.P., Luu, A.T., Laurent, T., Bengio, Y., Bresson, X.: Graph neural networks with learnable structural and positional representations. In: International Conference on Learning Representations (2022), https://openreview.net/forum?id=wTTjnvGphYj
13. Hersche, M., Karunaratne, G., Cherubini, G., Benini, L., Sebastian, A., Rahimi, A.: Constrained few-shot class-incremental learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 9057–9067 (2022)
14. Hinojosa, C., Marquez, M., Arguello, H., Adeli, E., Fei-Fei, L., Niebles, J.C.: Privhar: Recognizing human actions from privacy-preserving lens. In: European Conference on Computer Vision. pp. 314–332. Springer (2022)
15. Hou, S., Pan, X., Loy, C.C., Wang, Z., Lin, D.: Learning a unified classifier incrementally via rebalancing. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 831–839 (2019)
16. Jia, M., Tang, L., Chen, B.C., Cardie, C., Belongie, S., Hariharan, B., Lim, S.N.: Visual prompt tuning. In: European Conference on Computer Vision. pp. 709–727. Springer (2022)

17. Kirkpatrick, J., Pascanu, R., Rabinowitz, N., Veness, J., Desjardins, G., Rusu, A.A., Milan, K., Quan, J., Ramalho, T., Grabska-Barwinska, A., et al.: Overcoming catastrophic forgetting in neural networks. Proceedings of the national academy of sciences **114**(13), 3521–3526 (2017)
18. Kumawat, S., Nagahara, H.: Privacy-preserving action recognition via motion difference quantization. In: European Conference on Computer Vision. pp. 518–534. Springer (2022)
19. Lacoste, A., Luccioni, A., Schmidt, V., Dandres, T.: Quantifying the carbon emissions of machine learning. arXiv preprint arXiv:1910.09700 (2019)
20. Lester, B., Al-Rfou, R., Constant, N.: The power of scale for parameter-efficient prompt tuning. arXiv preprint arXiv:2104.08691 (2021)
21. Li, M., Xu, X., Fan, H., Zhou, P., Liu, J., Liu, J.W., Li, J., Keppo, J., Shou, M.Z., Yan, S.: Stprivacy: Spatio-temporal privacy-preserving action recognition. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 5106–5115 (2023)
22. Li, T., Ke, Q., Rahmani, H., Ho, R.E., Ding, H., Liu, J.: Else-net: Elastic semantic network for continual action recognition from skeleton data. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 13434–13443 (2021)
23. Li, X.L., Liang, P.: Prefix-tuning: Optimizing continuous prompts for generation. arXiv preprint arXiv:2101.00190 (2021)
24. Li, Y., Si, S., Li, G., Hsieh, C.J., Bengio, S.: Learnable fourier features for multi-dimensional spatial positional encoding. Advances in Neural Information Processing Systems **34**, 15816–15829 (2021)
25. Li, Z., Hoiem, D.: Learning without forgetting. IEEE transactions on pattern analysis and machine intelligence **40**(12), 2935–2947 (2017)
26. Liu, X., Yu, H.F., Dhillon, I., Hsieh, C.J.: Learning to encode position for transformer with continuous dynamical model. In: International conference on machine learning. pp. 6327–6335. PMLR (2020)
27. Ma, N., Zhang, H., Li, X., Zhou, S., Zhang, Z., Wen, J., Li, H., Gu, J., Bu, J.: Learning spatial-preserved skeleton representations for few-shot action recognition. In: European Conference on Computer Vision. pp. 174–191. Springer (2022)
28. Mallya, A., Lazebnik, S.: 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)
29. Mialon, G., Chen, D., Selosse, M., Mairal, J.: Graphit: Encoding graph structure in transformers. arXiv preprint arXiv:2106.05667 (2021)
30. Park, J.J., Florence, P., Straub, J., Newcombe, R., Lovegrove, S.: Deepsdf: Learning continuous signed distance functions for shape representation. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 165–174 (2019)
31. Peng, C., Zhao, K., Wang, T., Li, M., Lovell, B.C.: Few-shot class-incremental learning from an open-set perspective. In: European Conference on Computer Vision. pp. 382–397. Springer (2022)
32. Pernici, F., Bruni, M., Baecchi, C., Turchini, F., Del Bimbo, A.: Class-incremental learning with pre-allocated fixed classifiers. In: 2020 25th International Conference on Pattern Recognition (ICPR). pp. 6259–6266. IEEE (2021)
33. Radford, A., Kim, J.W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al.: Learning transferable visual models from natural language supervision. In: International conference on machine learning. pp. 8748–8763. PMLR (2021)

34. Razdaibiedina, A., Mao, Y., Hou, R., Khabsa, M., Lewis, M., Almahairi, A.: Progressive prompts: Continual learning for language models. arXiv preprint arXiv:2301.12314 (2023)
35. Rebuffi, S.A., Bilen, H., Vedaldi, A.: Efficient parametrization of multi-domain deep neural networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 8119–8127 (2018)
36. Ren, B., Liu, M., Ding, R., Liu, H.: A survey on 3d skeleton-based action recognition using learning method. Cyborg and Bionic Systems (2020)
37. Ridnik, T., Ben-Baruch, E., Noy, A., Zelnik-Manor, L.: Imagenet-21k pretraining for the masses. arXiv preprint arXiv:2104.10972 (2021)
38. Rusu, A.A., Rabinowitz, N.C., Desjardins, G., Soyer, H., Kirkpatrick, J., Kavukcuoglu, K., Pascanu, R., Hadsell, R.: Progressive neural networks. arXiv preprint arXiv:1606.04671 (2016)
39. Shahroudy, A., Liu, J., Ng, T.T., Wang, G.: Ntu rgb+ d: A large scale dataset for 3d human activity analysis. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 1010–1019 (2016)
40. Smedt, Q.D., Wannous, H., Vandeborre, J.P., Guerry, J., Saux, B.L., Filliat, D.: 3D Hand Gesture Recognition Using a Depth and Skeletal Dataset. In: Pratikakis, I., Dupont, F., Ovsjanikov, M. (eds.) Eurographics Workshop on 3D Object Retrieval. The Eurographics Association (2017). https://doi.org/10.2312/3dor.20171049
41. Smith, J.S., Karlinsky, L., Gutta, V., Cascante-Bonilla, P., Kim, D., Arbelle, A., Panda, R., Feris, R., Kira, Z.: 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)
42. Tang, Y.M., Peng, Y.X., Zheng, W.S.: When prompt-based incremental learning does not meet strong pretraining. arXiv preprint arXiv:2308.10445 (2023)
43. Tao, X., Hong, X., Chang, X., Dong, S., Wei, X., Gong, Y.: Few-shot class-incremental learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 12183–12192 (2020)
44. Van Den Oord, A., Vinyals, O., et al.: Neural discrete representation learning. Advances in neural information processing systems **30** (2017)
45. Villa, A., Alcázar, J.L., Alfarra, M., Alhamoud, K., Hurtado, J., Heilbron, F.C., Soto, A., Ghanem, B.: Pivot: Prompting for video continual learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 24214–24223 (2023)
46. Wang, X., Zhang, S., Qing, Z., Gao, C., Zhang, Y., Zhao, D., Sang, N.: Molo: Motion-augmented long-short contrastive learning for few-shot action recognition. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 18011–18021 (2023)
47. Wang, Y., Huang, Z., Hong, X.: S-prompts learning with pre-trained transformers: An occam's razor for domain incremental learning. Advances in Neural Information Processing Systems **35**, 5682–5695 (2022)
48. Wang, Z., Zhang, Z., Ebrahimi, S., Sun, R., Zhang, H., Lee, C.Y., Ren, X., Su, G., Perot, V., Dy, J., et al.: Dualprompt: Complementary prompting for rehearsal-free continual learning. In: European Conference on Computer Vision. pp. 631–648. Springer (2022)
49. Wang, Z., Zhang, Z., Lee, C.Y., Zhang, H., Sun, R., Ren, X., Su, G., Perot, V., Dy, J., Pfister, T.: Learning to prompt for continual learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 139–149 (2022)

50. Williams, W., Ringer, S., Ash, T., MacLeod, D., Dougherty, J., Hughes, J.: Hierarchical quantized autoencoders. Advances in Neural Information Processing Systems **33**, 4524–4535 (2020)
51. Yang, Y., Yuan, H., Li, X., Lin, Z., Torr, P., Tao, D.: Neural collapse inspired feature-classifier alignment for few-shot class-incremental learning. In: The Eleventh International Conference on Learning Representations (2023), https://openreview.net/forum?id=y5W8tpojhtJ
52. Yue, R., Tian, Z., Du, S.: Action recognition based on rgb and skeleton data sets: A survey. Neurocomputing (2022)
53. Zhang, H.B., Zhang, Y.X., Zhong, B., Lei, Q., Yang, L., Du, J.X., Chen, D.S.: A comprehensive survey of vision-based human action recognition methods. Sensors **19**(5), 1005 (2019)
54. Zheng, C., Vedaldi, A.: Online clustered codebook. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 22798–22807 (2023)
55. Zhou, D.W., Wang, F.Y., Ye, H.J., Ma, L., Pu, S., Zhan, D.C.: Forward compatible few-shot class-incremental learning. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 9046–9056 (2022)
56. Zhu, A., Ke, Q., Gong, M., Bailey, J.: Adaptive local-component-aware graph convolutional network for one-shot skeleton-based action recognition. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision. pp. 6038–6047 (2023)
57. Zhu, B., Niu, Y., Han, Y., Wu, Y., Zhang, H.: Prompt-aligned gradient for prompt tuning. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 15659–15669 (2023)

Supplementary Materials

POET: Prompt Offset Tuning for Continual Human Action Adaptation

Prachi Garg[1], Joseph K J[3], Vineeth N Balasubramanian[3], Necati Cihan Camgoz[2], Chengde Wan[2], Kenrick Kin[2], Weiguang Si[2], Shugao Ma[2], and Fernando De La Torre[1]

[1] Carnegie Mellon University, USA
[2] Meta Reality Labs
[3] Indian Institute of Technology, Hyderabad

In this supplementary material, we provide the following additional information which we could not include in the main paper due to space constraints. We provide all implementation details (besides the code) herein.

Table of Contents

A Implementation Details

A.1 Training Details

We use the same hyperparameters across all experiments for the NTU RGB+D activity recognition benchmark (in Tables 1, 3, 4 of the main paper, and Supplementary Table 1 and Figure 6). In the SHREC 2017 gesture benchmark also, all our experiments follow the exact same hyperparameter combination and learning strategy in Table 2 and Figure 7.

Activity recognition on NTU RGB+D benchmark. In the base session $\mathcal{UB}^{(0)}$, we train the CTR-GCN [8] backbone for 50 epochs with initial LR=0.1. We use a batch size of 64 as in the original paper. Every continual user session $\mathcal{US}^{(t)}$ is trained for 5 epochs with an initial LR=0.1.

Gesture recognition on SHREC 2017. We train DG-STA [7] model in the base session $\mathcal{UB}^{(0)}$ for 300 epochs and initial LR=0.001, using batch size 32 and dropout set to 0.2 (default hyperparameters from the DG-STA paper). It is updated for 30 epochs in each user session $\mathcal{US}^{(t)}$, starting with initial LR=0.01.

We select higher initial LRs in continual sessions $t > 0$ because starting with a lower learning rate as compared to base session (as is standard practice in continual learning to prevent catastrophic forgetting) renders limited plasticity and the model is completely unable to learn new knowledge. Our choice enables learning of new knowledge from the few user samples, and we can study the model's stability-plasticity trade-offs, optimizing for a balance between the two. The continual session learning rates given above are used to update the (i) the classifier f_c, (ii) selected prompts in $\mathbf{P}$, and (iii) selected keys in $\boldsymbol{K}$. But for updating query adaptor f_{QA}, we use a learning rate of 0.01, in $t > 0$ for both benchmarks. At the same time, we freeze all other layers in the query model. We find this adapts query adaptor to new tasks without overwriting existing base knowledge. In each continual session $t > 0$, we use a batch size of 25 for NTU RGB+D, as there are 5 new classes each having 5 training samples (single batch per epoch). Similarly, batch size is 10 for SHREC 2017 with 2 new classes with 5 training samples each. For the clustering loss coefficient in Eq. 6, we use $\lambda = 0.1$ for all experiments.

A.2 Base Session $\mathcal{UB}^{(0)}$: Prompt Instantiation and Training

Prompt instantiation, CTR-GCN: CTR-GCN is a spatio-temporal graph convolutional network architecture with 10 multi-scale temporal convolutional (TCN-GCN) layers followed by an average pool over the spatial and temporal dimensions, and a final linear classification layer. The output feature dimensionality of the first four layers (L1-L4) is 64 channels, next four (L5-L8) is 128 and final two layers (L9 and L10) have 256 channels. An input skeleton sequence has 64 temporal frames, each consisting of 25 body joints, such that $x \in \mathcal{R}^{64\times25\times3}$.

The input embedding after layer L1 is $x_e \in \mathcal{R}^{64\times25\times64}$, such that $C_e = 64$. We start from a prompt pool $\mathbf{P} = \{\boldsymbol{P_j}\}_{j=1}^{T}$ of size $M = T = 64$. Each prompt in

the pool $P_j \in \mathcal{R}^{25\times64}$ is designed to match the spatial and feature dimensions of the input embedding x_e. There are $M = 64$ keys, each having feature dimension $\boldsymbol{k_j} \in \mathcal{R}^{64}$. Query adaptor f_{QA} maps a feature embedding of size 256 (from the last layer of query model f'_g) to $C_e = 64$ for feature dimension compatibility with the keys and prompts. We select $T = 64$ prompts from the pool. After instantiating the prompt and key parameters, we train the prompt pool $\mathbf{P}$, keys $\boldsymbol{K}$, query adaptor f_{QA}, along with all main model parameters f_g, f_c, f_e on the base session data $\mathcal{D}^{(0)}$ as per Algorithm 2.

Prompt instantiation, DG-STA: DG-STA is a fully connected graph transformer architecture with multi-head spatial and temporal attention layers. For every input skeleton hand gesture sequence, we use 8 temporal frames, each having 22 hand joint coordinates such that input is $x \in \mathcal{R}^{8\times22\times3}$. Output of the transformer's input embedding layer f_e is $x_e \in \mathcal{R}^{8\times22\times128}$. As DG-STA expects a fully connected spatio-temporal graph across all joints in all frames, this is reshaped to a size $x_e \in \mathcal{R}^{176\times128}$ before passing it to the first attention layer of the transformer. We add our prompt to this reshaped embedding. We start from a pool of size $M = 8$ prompts. As DG-STA is a transformer architecture, the output feature dimensionality remains constant (at 128) and the query adaptor input and output dimensions are the same ($C_e = 128$). The base session here is also trained using Algorithm 2.

A.3 Classifier Update Protocol in $\mathcal{US}^{(t)}, t > 0$

Existing related work such as in few-shot class-incremental learning learn non-parametric classifiers by extracting class-mean prototypes, and do not expand the classifier with any new weights. However, we need to expand and train the classifier in order to obtain the error gradients for updating the prompts. We

Algorithm 2 Initialization & Training of Prompts, $t = 0$

Input: Model $f^P(.) = f_c^P \circ f_g^P \circ f_e^P(.)$, *pretrained* only on base $\mathcal{UB}^{(0)}$ data.
Initialize:
1. Main model f as: $f_e \leftarrow f_e^P, f_g \leftarrow f_g^P, f_c \leftarrow f_c^P$.
2. Prompt pool $\mathbf{P} = \{\boldsymbol{P_j}\}_{j=1}^{T}$, Keys $\boldsymbol{K} = \{\boldsymbol{k_j}\}_{j=1}^{T}$ from $\mathcal{U}(0, 1)$.
2. Query function model f_q as: $f'_e \leftarrow f_e^P, f'_g \leftarrow f_g^P$. f_{QA} is randomly initialized.

Freeze: Query function layers f'_g, f'_e.

for epochs and batch in base dataset $(\mathbf{X}_i^0, \mathbf{y}_i^0)_{i=1}^{|\mathcal{D}^{(0)}|}$ do
1. Get query feature $\boldsymbol{q}$ (Eq. 4) ; Compute $\gamma(.)$ b/w query $\boldsymbol{q}$ and keys $\boldsymbol{K}$
2. Sort $\gamma(.)$; Get ordered key index sequence $(s_i)_{i=1}^{T}$ (Eq. 7)
3. Read pool memory $\mathbf{P}$ in order $(s_i)_{i=1}^{T} \rightarrow$ Get prompt offsets $\mathbf{P_T}$
4. Get $\mathbf{X_e}$; **Add** $\mathbf{P_T}$ to it (Eq. 8); get prediction $\mathbf{y}$ from prompted input (Eq. 1)
5. Use cross entropy loss (Equation 5) to **update** prompt associated parameters $f_{QA}, \boldsymbol{K}, \mathbf{P}$ and all main model parameters f_g, f_c, f_e.
6. Use clustering loss (Equation 6) to **update** f_{QA}, $\boldsymbol{K}$.

end
Freeze: Main model feature extractor f_g, input embedding layer f_e for time $t > 0$.

Table 1: Experimenting with different classifiers for NTU RGB+D. Here, we report the task-specific accuracy for each task the model has learnt so far, after learning every new task. Notice the sharp **forgetting of the *intermediate* 'New-Old'** tasks in regular classifier and our improvement using a cosine normalized classifier. Note, we are using a dynamically expanding parametric classifier in all experiments.

	$\mathcal{US}^{(0)}$	$\mathcal{US}^{(1)}$		$\mathcal{US}^{(2)}$			$\mathcal{US}^{(3)}$				$\mathcal{US}^{(4)}$					
Activity	Base	$\mathcal{US}^{(0)}$	$\mathcal{US}^{(1)}$	$\mathcal{US}^{(0)}$	$\mathcal{US}^{(1)}$	$\mathcal{US}^{(2)}$	$\mathcal{US}^{(0)}$	$\mathcal{US}^{(1)}$	$\mathcal{US}^{(2)}$	$\mathcal{US}^{(3)}$	$\mathcal{US}^{(0)}$	$\mathcal{US}^{(1)}$	$\mathcal{US}^{(2)}$	$\mathcal{US}^{(3)}$	$\mathcal{US}^{(4)}$	**Avg**
Regular, Freeze	89.2	73.5	**72.5**	71.3	2.1	**63.8**	65.3	0.0	0.0	52.9	58.7	0.0	0.0	0.0	48.9	43.1 (-16.1)
Regular, Tune	89.2	80.9	64.9	67.4	22.1	34.4	57.2	6.32	24.3	26.9	45.0	4.6	**20.9**	13.3	21.4	35.2 (-24.0)
Cosine (Ours)	87.9	**84.9**	65.3	**83.2**	**56.0**	45.8	**78.2**	**36.3**	**34.5**	**60.0**	**71.3**	**18.5**	19.8	**46.2**	**57.4**	**59.2**

observe that a key source of forgetting is from the classifier as the logits tend to become biased towards the few-training samples of new classes. For SHREC/DG-STA, we observe that fine-tuning the entire classifier leads to a significant drop in performance of old classes (experiment 'FE' in Table 2 of main paper). This is because the frozen backbone is trained only on 1146 training samples from the 8 base session classes. Hence, the SHREC backbone exhibits low stability for retaining old knowledge, when updated on new data in subsequent sessions $t > 0$. To alleviate this, we use a simple classifier update trick wherein after expanding the classifier in every continual session $t > 0$, we turn the gradients of the old parameters in the classifier to zero before the error backpropagation. This trick has also been shown to work in prior continual prompt tuning works L2P [49], CODA-P [41].

We observe that this trick proves sub-optimal in the NTU RGB+D (CTR-GCN) dataset. In Table 1, we evaluate performance on each usr session individually after learning every new user session. We observe that both, freezing ('**Regular, Freeze**') or finetuning ('**Regular, Tune**') of old class parameters in classifier suffers from poor stability-plasticity trade-offs. *We observe that the intermediate tasks ($\mathcal{US}^{(1)}$, $\mathcal{US}^{(2)}$, $\mathcal{US}^{(3)}$) learnt using few shot data particularly suffer severe forgetting as soon as the next task is learnt.* This is not the case for $\mathcal{US}^{(0)}$ performance in incremental sessions $t > 0$ because the base feature extractor is trained on sufficient data in the activity benchmark (26,731 samples) and frozen henceforth, retaining performance on the base task $\mathcal{US}^{(0)}$ even in $t > 0$. We call this the **'New-old forgetting'**. To address the biasing of logits towards new classes, we replace the main model's linear classification layer f_c with a cosine normalization classifier θ_c^T as:

$$p_{(x)} = \frac{exp(\eta < \theta_{c_i}^T f_g(x) >)}{\Sigma_j exp(\eta < \theta_{c_j}^T f_g(x) >)} \tag{1}$$

where η is a learnable scale parameter we learn in the base session and freeze in subsequent sessions. Also, in incremental sessions, we initialize the new-class parameters in the classifier as mean of previous class parameters. Notice the significant boost in performance across all tasks $\mathcal{US}^{(i)}$ in the cosine normalized classifier in Table 1.

A.4 Analyzing Where to Attach Prompts

We study the impact of attaching our prompt offsets at different main model layers in CTR-GCN in Figure 1. As mentioned in Sec A.2, output feature dimensionality (number of channels) of the first four layers in CTR-GCN is the same, $C_e = 64$. We desire that the feature dimension size of the (i) prompt parameters, (ii) key parameters, and (iii) query adaptor be consistent with the main model feature embedding dimension at the layer being prompted. Hence, we only prompt the first four layers for a fair comparison as results may vary with variation in size of feature dimension being prompted. We created a separate 30% validation set from the training data of $t \geq 1$ classes for this analysis. Our findings in Figure 1 indicate that the highest 'New' task performance at the end of all four user sessions is achieved by prompting layer L1. We select layer L1 across all experiments in the paper.

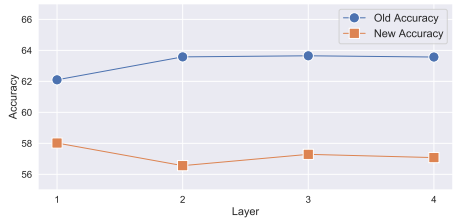


Fig. 1: Empirical analysis to study the impact of the layer at which our prompt is attached. Y-axis shows 'Old' and 'New' classes accuracy after Task 4 (after learning all 60 classes). We add a prompt of size $P_{T'} \in \mathcal{R}^{64,25,64}$ to different layers $\{1, 2, 3, 4\}$ of CTR-GCN, evaluated on the NTU RGB+D validation set. We select layer L1 due to its high performance on new classes.

A.5 Additional Dataset Details

The NTU RGB+D dataset has been collected from Microsoft Kinect V2 sensors from three different camera viewpoints by annotating 40 human users. The 60 classes consist of 40 daily action categories (drinking water, reading, writing, etc.), 9 health-related actions (coughing, sneezing, headache, etc.), and 11 mutual actions (handshaking, pushing another person, walking towards another person, etc.). NTU RGB+D has 40,320 training and 16,560 testing samples across 60 classes. We use the first 40 daily action classes as the base model data, and update on $5 \times 5 = 25$ training samples in each of the 4 incremental session. The base model is trained on 26731 training samples from the 40 base classes.

The SHREC 2017 dataset has 14 fine-grained hand gestures captured using the short-range Intel Real Sense Depth, from 28 human subjects in second-person view. There are 1980 training and 840 testing samples. The base model for this

benchmark is trained on 1146 training samples from 8 classes, and updated on $2 \times 5 = 10$ training samples in each of the 3 incremental session.

Given the relatively small sizes of both datasets, we follow a class-incremental setting in our work, viz., our user sessions include new classes over time, not necessarily new users. Evaluating our work on user-specific continual streams is left as future work for larger datasets where this is feasible.

Class splits in user sessions. There are two existing continual benchmarks for 3D skeleton-based action recognition: The experimental protocol of NTU RGB+D based class incremental benchmark [22] involves a single continual session, learning 50 classes in the base session, and 10 new classes in a single-incremental session. We consider this to be limiting and too simplistic to study our problem setting on. Moreover, their code is not publicly available. On the other hand, the more recent data-free class incremental learning for hand gesture benchmark [1] learns 8 classes in base session and updates the model on a single class over 6 incremental sessions. We believe that this too does not lend itself to our setting, when there are only 5-shots for training each continual class and the base model is trained on small-scale data.

A.6 Adaptation of Baselines to Problem Setting

Adapting Learning to Prompt (L2P): We experiment with selecting $T' = 4$ and $T' = 64$ (same as POET) for this comparison on CTR-GCN backbone. We find the results with 4 prompts marginally better, hence we report those in Table 3 of main paper. We experiment with both temporal concatenation and spatio-temporal concatenation followed by remapping. For DG-STA, we select 8 prompts from a pool of size $M = 10$, each prompt (22, 128); and concatenate this prompt of size (8, 22, 128) along the spatio-temporal dimension 176 of the input embedding (176, 128). We map this back to (176, 128) using a fully connected layer. In experiments where we remap using FC layer, we update this layer as well in future incremental sessions. To update the expanded classifier, we make logits of previous classes -inf, same as the classifier training protocol used in L2P, Dual-P and CODA-P.

Adapting CODA-P: For activity recognition on CTR-GCN backbone, we construct a 4 dimensional CODA-prompt of size (100, T', 25, 64), such that the prompt component dimension of 100 gets collapsed after weighing and we can concatenate prompt of size (T', 25, 64) along the temporal or rolled out spatio-temporal dimension (same as L2P). The size of memory buffer (T') is kept consistent with L2P experiments.

For gesture recognition on DG-STA backbone, [41] tunes a ViT-B/16 pre-trained on ImageNet-1K architecture instead of prompt tuning. This refers to concatenating half of the prompt to K and V of the MSA layer instead of concatenating along the token dimension. However, we don't have the luxury to modify the input embedding size or assume that the backbone is a transformer. Also for a fair comparison with L2P, we concatenate a fixed sized prompt to the input embedding and use a fully connected layer to map the feature dimension

back to default input embedding size of (176, 128). Hence, sizes are as follows: initial prompt size (100, 5, 128), Attention (100, 128), Key (100, 128), each have 100 prompt components. After alpha weighting, fixed sized prompt P (5, 128) gets concatenated along joint dimension. We don't update query adaptor in this experiment. We implement all loss functions, including the orthogonality loss as is. Also, note that we attach prompts only at the input embedding layer for fair comparison of the prompting strategy.

Adapting ALICE [31]: For training the base session, we add a projection head to the feature extractor before the classification layer. Like the paper, we use two augmentations of every input and losses from the two augmentations are averaged before backpropagation. We use angular penalty for training the classifier. After base session learning, the projection head and classification layer are discarded, only learning feature extractor. Next, we use cosine distance and class-wise mean to generate class-prototypical feature vectors from the feature extractor's output. These prototypes are used for nearest mean classification. For incremental steps, no training is done. Only new class means are computed and evaluation is performed.

Adapting LUCIR [15] and EWC [17]: We initialize model from previous checkpoint, such that classifier has random weights for new classes and previous classifier weights are copied over to previous class parameters in the classifier. Cross entropy loss is computed between logits from all the classes and current task ground truth labels. All regularization loss terms are implemented as proposed in their respective papers. For LwF [25], we use a $\lambda = 1.0$.

B Additional Results

B.1 POET's Effectiveness in Learning New Knowledge while Mitigating Catastrophic Forgetting

Table 2: POET Ablation Table: We exhaustively study the contribution of various POET components towards mitigating forgetting of old knowledge (Old) as well as learning new knowledge (New).

	Prompt	Prompt Selection	Prompt Integration	$\mathcal{UB}^{(0)}$	$\mathcal{US}^{(4)}$			
Ablation			(Operator, #Prompts)	Base (↑)	Old (↑)	New (↑)	Avg (↑)	A_{HM} (↑)
POET (Ours)	✓	✓	ADD T	87.9	57.2 ± 1.0	**55.8** ± 5.9	**57.1** ± 1.1	**56.3** ± 3.2
Importance of prompts in POET								
POET w/o **Prompts**	✗			87.9	60.8 ± 0.5 (+3.6)	18.4 ± 1.0 (-37.4)	57.3 ± 0.4	28.3 ± 1.1
POET w/o **C.U.P.**	✓	✓	ADD T	89.2	45.5 ± 1.4 (-11.7)	53.6 ± 3.7 (-2.2)	46.2 ± 1.2	49.1 ± 1.5
POET w/o **{Prompts, C.U.P.}**	✗			88.4	40.0 ± 1.6 (-17.2)	51.0 ± 2.3 (-4.8)	44.8 ± 1.1	40.9 ± 1.4
POET w/o **{Prompts, C.U.P., Freezing}**	✗			88.4	0.2 ± 0.5 (-57.0)	36.0 ± 10.1 (-19.8)	3.2 ± 0.8	0.3 ± 1.0
Impact of Ordered Selection								
POET w/o **Ordered Selection**	✓	✓	ADD T	88.2	59.9 ± 1.1 (+2.7)	52.5 ± 4.2 (-3.2)	59.3 ± 0.9	55.9 ± 2.2
Relative importance of our Prompt Selection versus Prompt Integration								
POET Selection w/o **Addition**	✓	✓	Cross-Attend T	82.9	57.0 ± 2.0 (-0.2)	31.0 ± 4.8 (-24.8)	54.9 ± 1.8	39.9 ± 4.0
POET Addition w/o **Selection**	✓	Standalone	ADD T	88.6	58.8 ± 3.2 (+1.6)	54.0 ± 3.4 (-1.8)	58.4 ± 1.2	56.2 ± 2.2
POET Addition w/o **Selection**	✓	Standalone	ADD 1	88.2	56.9 ± 1.1 (-0.3)	54.7 ± 5.3 (-1.1)	56.7 ± 1.2	55.7 ± 3.1
POET w/o **{Selection, Addition}**	✓	Standalone	Cross-Attend T	43.3	25.6 ± 1.3 (-31.6)	5.0 ± 4.1 (-50.8)	23.9 ± 1.2	7.8 ± 4.9

How Well Does POET Mitigate Catastrophic Forgetting? In this section we dive deeper into understanding how our proposed Prompt Offset Tuning methodology *mitigates catastrophic forgetting* in our few-shot class incremental setting for action recognition. We report mean and standard deviation across 10 experimental runs (10 sets of few-shots) for robustness ablation. The 'Old' only accuracy gives a direct comparison of the forgetting.

Our classifier expands in each continual session and fine-tuning the entire classifier on cross entropy loss of new classes leads to erosion of previous class weights in the classifier. **'C.U.P.'** refers to our classifier update protocol used to prevent forgetting from the classifier. We use a cosine normalized classifier for NTU RGB+D dataset (see Sec. A.3 which helps mitigate forgetting from the classifier as shown by the 11.7% $\downarrow$ in 'Old' class performance of **'POET w/o C.U.P.'**, w.r.t. POET.

In **'POET w/o Prompts'**, we simply remove our prompts altogether, keeping the cosine normalized classifier to observe the prompt only affect. While Old performance is slightly better, the backbone doesn't learn new knowledge as shown by 37.4% $\downarrow$ in 'New' class performance. If we remove both prompts and classifier update protocol **'POET w/o Prompts, C.U.P.'**, we get vanilla Feature Extraction without any freezing or regularizing of classifier weights. It suffers in both Old and New classes. Further, as highlighted in the main paper as well, freezing our backbone during the continual user sessions $t > 0$ is of prime importance given our few-shot setting where the overfitting to few training samples exacerbates the overwriting of existing knowledge, leading to a complete washout of previous knowledge as seen by the 0.2 Old performance.

Notice, in the absence of our ordered prompt selection, New performance suffers by 3.3% as the same prompts are selected and updated everytime. In POET, we ensure the prompts get selected in the right temporal sequence, learning new temporal semantics for new classes hence enabling better adaptation to new classes.

Finally, within prompts, we replace our prompt selection mechanism completely by standalone T prompts or a single prompt. This shows how POET's spatio-temporal temporally consistent selection mechanism helps learn new orderings of T prompts as compared to attaching prompts without selecting them using an input dependent query. We also use cross attention along the temporal dimension as attachment operator $f_p(.)$ and find addition consistently outperforms. **'POET w/o Selection, Addition'** experiment shows the importance of our design choices as unsuitable selection and integration mechanisms can be fatal to continual learning performance.

Backward Forgetting Metric (BWF): In addition to these results, we report the average forgetting metric [5, 6] after the model has been trained continually on all user sessions (after $\mathcal{US}^{(4)}$) as:

$$F_k = \frac{1}{k-1} \sum_{j=1}^{k-1} f_j^k \tag{2}$$

Table 3: Backward Forgetting Metric (%, ↓): Here we present BWF after user session $\mathcal{US}^{(4)}$. POET significantly mitigates forgetting on old classes.

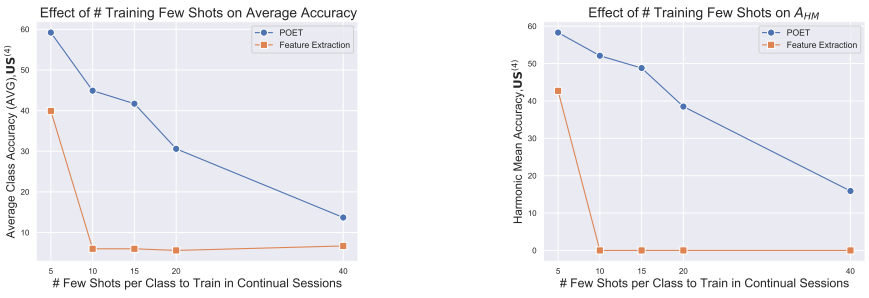
Method	$\mathcal{US}^{(4)}$ Forgetting (↓)
FE+Replay	-0.90 ± 0.65
FE	52.83 ± 2.09
FT	54.08 ± 9.17
FE, Freeze	34.02 ± 0.83
POET (Ours)	29.91 ± 0.34

where f_j^k is the forgetting on previous task 'j' after the model is trained with all the few-shots up till task k:

$$f_j^k = \max_{l \in \{1,\ldots,k-1\}} a_{l,B_{l,j}} - a_{k,B_{k,j}} \tag{3}$$

In effect, this is the same as difference in performance of each previous task 'j' at the end of $\mathcal{US}^{(4)}$ from when the task was first introduced (*New* in $\mathcal{US}^{(j)}$).

Role of number of few-shots in continual learning: We also vary the number of few shots used for training per new class in continual sessions in Fig 2. The Feature Extraction baseline reduces to zero A_{HM} because the Old class performance reduces to zero. We find our prompt offsets in POET (=FE+Prompts) significantly help retain old class performance as compared to Feature Extraction, without any explicit forgetting measure due to our ordered prompt selection and clustering loss. It can be noted that our method is particularly well suited for very few training samples per user (<15) and may require additional explicit regularization or freezing of prompt pool to mitigate forgetting for large number of training samples (>20).



(a) Average Accuracy (AVG) **(b)** A_{HM}

Fig. 2: Effect of variation in number of few-shot samples used for training in user sessions $\mathcal{US}^{(1)}$-$\mathcal{US}^{(4)}$ on stability-plasticity trade-offs in our few-shot continual setting.

B.2 Impact of Prompts in POET

In Figures 3 and 4, we qualitatively study the impact of prompts by removing prompts from POET (the corresponding feature extraction baselines 'FE' for NTU RGB+D and 'FE++' for SHREC). Prior continual learning works rely on ImageNet21K pretrained ViT [37] (L2P [49], Dual-P [48], CODA-P [41]) or WebImageText pretrained CLIP model [33] (S-Prompts [47], PIVOT [45]) for prompt tuning. In Fig. 5, we show the significant disparity in scale of pretraining dataset as we use only base class dataset from the benchmark itself for pretraining and every new user session sees a non-overlapping set of classes. Despite of this, POET shows promising results.

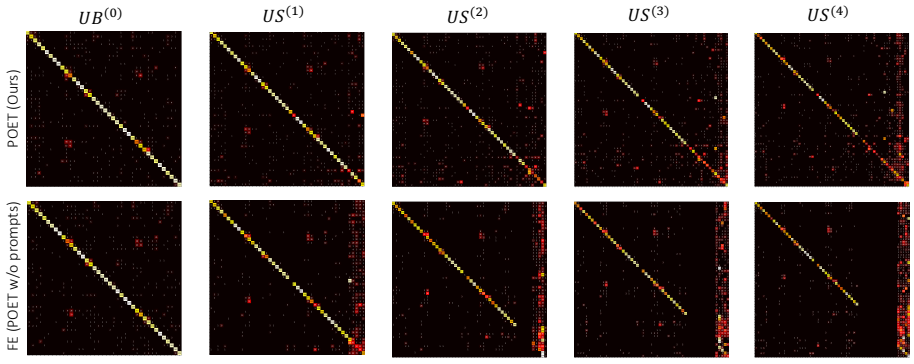


Fig. 3: *Confusion matrices showing the impact of our prompt offsets across 4 user sessions in* ***NTU RGB+D activity recognition benchmark.*** We compare confusion across new and old actions in POET and POET w/o prompts ablation. Starting from 40 default classes in $\mathcal{UB}^{(\prime)}$, we learn new classes {$\mathcal{US}^{(1)}$: sneeze, stagger, fall, touch head, touch chest}; {$\mathcal{US}^{(2)}$: touch back, touch neck, nausea, user fan, punch}; {$\mathcal{US}^{(3)}$: kick, push, pat back, point finger, hug}; {$\mathcal{US}^{(4)}$: give, touch pocket, handshake, walk towards, walk away}. Prompts enable retention of the intermediate **'New-Old'** classes very well, while FE gets heavily biased towards the new classes (see last 5 columns in each matrix).

B.3 Stability-Plasticity Trade-offs via New/Old performance

In Fig. 6, we study the average accuracy of only new classes (New) and only old classes (Old) after every user session in activity recognition benchmark. As stated in Sec 5 of main paper, we observe that (i) any method that does not freeze the backbone such as knowledge distillation (LWF and LUCIR), prior-based regularization methods (like EWC), or vanilla Fine-tuning baselines (FT) completely forget old performance from $\mathcal{US}^{(1)}$ itself. POET is short of only FE+Replay which is an upper bound. (ii) Any existing prompt tuning work

which does not update their query function such as L2P, CODA-P, APT in graph (B), is unable to learn new classes well.

In Fig. 7, we observe similar trends. Additionally, (i) ALICE retains old knowledge very well as it does not use a parametric classifier for incremental sessions. However, ALICE is unable to learn new classes well. We also observe that (ii) DG-STA backbone has very high plasticity when fine-tuned on new tasks (see New performance of FT and LWF in graph B). But these baselines while plastic, completely forget old classes. POET achieves the best stability-plasticity trade-offs (indicated by A_{HM} in main paper).

B.4 Robustness to class order in user sessions

The default continual order in which different gesture classes appear till now was: $\{\mathcal{US}^{(0)}$: Grab, Tap, Expand, Pinch, Rotate-CW, Rotate-CCW, Swipe-R, Swipe-L$\} \rightarrow \{\mathcal{US}^{(1)}$: Swipe-U, Swipe-D$\} \rightarrow \{\mathcal{US}^{(2)}$: Swipe-x, Swipe-+$\} \rightarrow \{\mathcal{US}^{(3)}$: Swipe-v, Shake$\}$. In Fig 8, we swap the base and incremental classes in SHREC benchmark to a new ordering: $\{\mathcal{US}^{(0)}$: Swipe-R, Swipe-L, Swipe-U, Swipe-D, Swipe-x, Swipe-+, Swipe-v, Shake$\} \rightarrow \{\mathcal{US}^{(1)}$: Grap, Tap$\} \rightarrow \{\mathcal{US}^{(2)}$: Expand, Pinch$\} \rightarrow \{\mathcal{US}^{(3)}$: Rotate-CW, Rotate-CCW$\}$. We find **'POET'** gives an $AVG = 57.3$ as compared to **'ALICE'**, $AVG = 55.9$ and **'FE, Freeze'**, $AVG = 55.3$ at the end of 3 user sessions even though our backbone is now trained on a different set of classes and we completely reversed the semantic order in which prompts learn different fine-grained gesture classes. This demonstrates robustness to variation in continual class order across tasks.

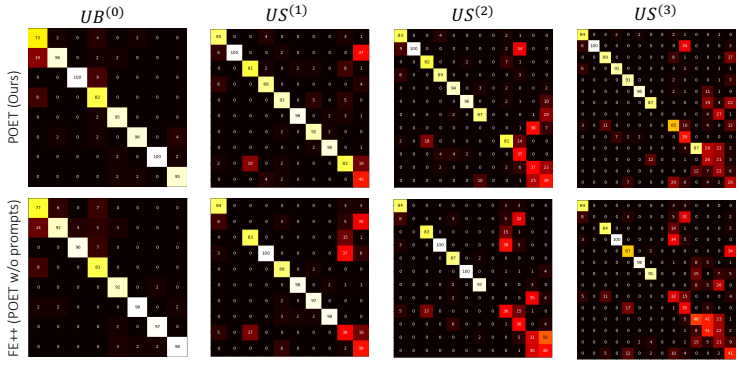


Fig. 4: *Confusion matrices showing the impact of our prompt offsets across 3 user sessions in* ***SHREC 2017 gesture recognition benchmark.*** $\mathcal{US}^{(0)}$ has hand gestures grab, tap, expand, pinch, rotate clockwise, rotate counter-clockwise, swipe right, swipe left$\}$. $\{\mathcal{US}^{(1)}$: swipe up, swipe down$\}$, $\{\mathcal{US}^{(1)}$: swipe-x, swipe-+$\}$, $\{\mathcal{US}^{(1)}$: swipe-v, shake$\}$. Even though the classes are fine-grained, the prompts help retain old class semantics well.

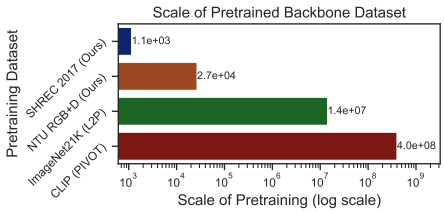


Fig. 5: Scale of pretraining used for the prompt tuning backbones. For (Our) benchmarks on NTU RGB+D and SHREC 2017, numbers represent the base class training data used. Our POETs continually learn new actions mitigating catastrophic forgetting, without massive pretraining, and only rely on prompts.

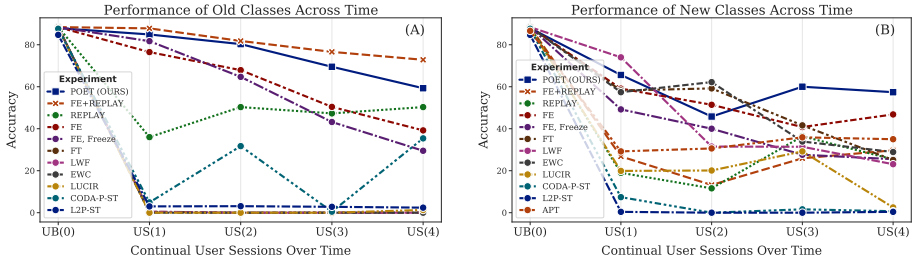


Fig. 6: Old and New class performance for NTU RGB+D.

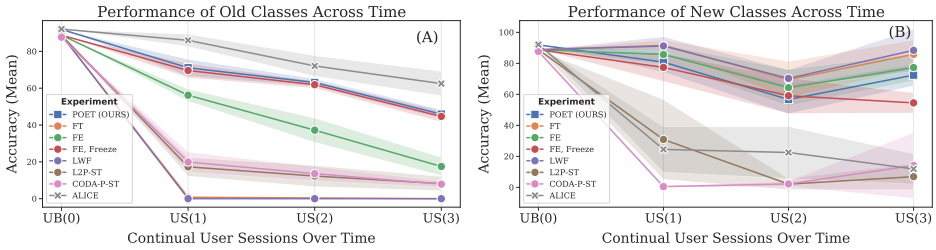


Fig. 7: Old and New class performance for SHREC 2017. Reporting Mean and STD over 5-sets of user few-shots.

B.5 Ordered Key Index Selection $(s_i)_{i=1}^{T}$: Qualitative Results

In Section 4 of main paper, we explained our sorted *ordered key index selection* for selecting temporally consistent prompts from the pool. In Figure 3 of main paper, we visualized $(s_i)_{i=1}^{T}$ ordering statistics at the task-level. In Figure 9, we investigate class-wise ordering statistics at inference time, performing inference after each continual task. We find that the ordering statistics are not only disparate for different classes, the statistics for a class remain consistent across continual tasks. The temporal discriminability in these studies further

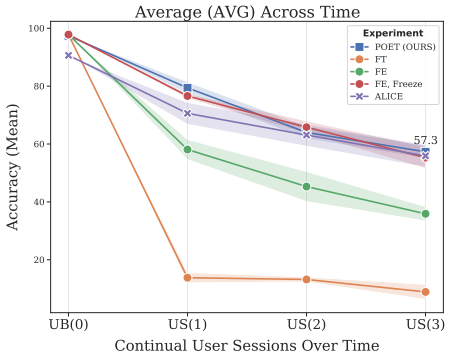


Fig. 8: Here, we change the set of classes in each session from the default order seen before. We report average accuracy of all classes learnt by the model after adding each new session. New ordering: $\{\mathcal{US}^{(0)}$: Swipe-R, Swipe-L, Swipe-U, Swipe-D, Swipe-x, Swipe-+, Swipe-v, Shake$\} \rightarrow \{\mathcal{US}^{(1)}$: Grap, Tap$\} \rightarrow \{\mathcal{US}^{(2)}$: Expand, Pinch$\} \rightarrow \{\mathcal{US}^{(3)}$: Rotate-CW, Rotate-CCW$\}$.

establishes that our learnable prompt selection mechanism is temporally with the 4D input skeleton. Finally, we also demonstrate instance-level statistics in Figure 10 and Figure 11 as our prompt mechanism is designed to select relevant (temporal ordered) prompts conditioned on every input instance. This means that our method does not depend on disparate task-wise or class-wise dataset splits and can even be used for online continual learning settings that do not have clear task boundaries.

B.6 Prompt Pool Expansion

We further present the order-preserving prompt pool expansion Algorithm 3 that (A) expands pool to learn new knowledge, (B) freezes previous prompts to prevent forgetting, and (C) forces usage of new prompts at the end of the sequence, hence alleviating the prompt pool collapse issue while preserving already existing temporal order statistics Figure 4 of main paper. We find $R = 6$ new prompts to be the best empirically for NTU RGB+D and $R = 2$ for SHREC 2017 using our 30% validation set of incremental sessions. Algorithm 3 presents our algorithm for prompt pool expansion.

C Broader Impact and Limitations

Privacy-aware human action recognition in extended reality devices: In order to protect users' privacy and security in head mounted devices, we incorporate privacy awareness in our continual learning setup: (i) By not storing any old class exemplar data or prototypes for replay in continual user sessions. All data is trained in a session and discarded henceforth. (ii) By using only 3D

Algorithm 3 Prompt Pool Expansion at Train Time, $t \geq 1$

Input: Query function f_q, keys $\boldsymbol{K} = \{\boldsymbol{k_j}\}_{j=1}^{T}$, prompt pool $\mathbf{P} = \{\boldsymbol{P_j}\}_{j=1}^{T}$; main model f_e, f_g, f_c
Expand:
Pool and keys by R new prompts as: $\mathbf{P}_M \rightarrow \mathbf{P}_{M+R}; \boldsymbol{K}_M \rightarrow \boldsymbol{K}_{M+R}$
Where $\mathbf{P}_{M+R} = \{\mathbf{P}_M; \mathbf{P}_R\}$ (attach new prompts at the end of existing tensor)
Initialize: New prompts $\mathbf{P}_i \leftarrow \mathcal{U}(0,1)$; new keys $\boldsymbol{K}_i \leftarrow Mean(\boldsymbol{K}_M)$
Construct $\mathbf{P}_T$ as:
1. Find $T-R$ key indices $\boldsymbol{K}_{T-R}$ using Eq. 7. Use this sequence to read previous prompts in the pool $\mathbf{P}_M$ and form $\mathbf{P}_{T-R}$.
2. Concatenate $\mathbf{P}_R$ new prompts at the end of the sequence: $\mathbf{P}_T = \{\mathbf{P}_{T-R}; \mathbf{P}_R\}$ (i.e. explicitly use R new prompts).
Freeze: Previous task prompts in the pool $\mathbf{P}_M$.
Train: New prompts $\mathbf{P}_R$, all keys $\boldsymbol{K}_{M+R}$ (to learn global inter-task selection), query adaptor f_{QA}, and classifier f_c.

skeleton joint input modality for action recognition, we circumvent the visual privacy violation and user identity revelation in video-based HAR [14,18,21]. While we are a privacy-aware continual learning setting, we do not claim differential privacy and studying differential privacy in our prompts will be an interesting future work direction.

Data-free adaptation of action models for new user categories: Our key motivation for a *data-free* prompt tuning-based action recognition model adaptation to new categories over time is to maintain privacy of past sessions' data. However, this also has other advantages. Firstly, a data-free solution does not require a memory budget on the edge device for replay of old class data. Secondly, it has become commonplace to have access to large pre-trained backbones, but there is limited knowledge and often lack of access to such a dataset for such pre-training. Finally, prompts via a vector-quantized prompt pool memory offer a compact, learnable and automatically retrievable bottleneck of task-specific information (like in auto-decoders). Even if businesses can store and retrain their model on all previous data to continually adapt to new data, training large models incurs high carbon footprints [19]. Prompts offer a cost-efficient and low carbon footprint solution to retraining large models from scratch every time new data of value becomes available. Evaluating our design choices to large pre-trained models for skeletal data, as and when they become available, is another direction of future work.

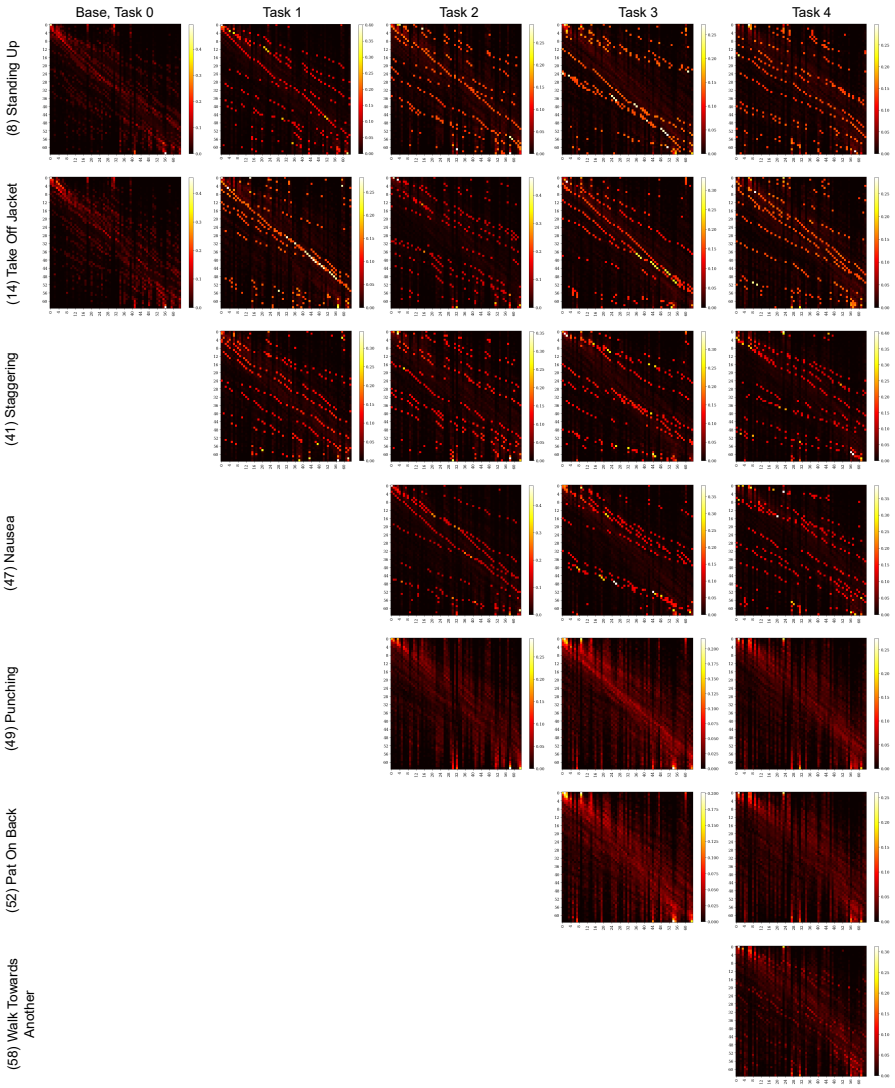


Fig. 9: Class-level Ordered Prompt Selection using POET, Task t is same as $\mathcal{US}^{(t)}$: Here, we analyse the ordered prompt selection statistics for our method for different classes at **test time**. For each class shown in column 1, we plot the prompt selection order at test time for each continual model checkpoint (starting from when that class was first introduced to the continual system and checking after updating the model on new classes each time). We **observe** that class-wise selection statistics are retained even after Task 4 (notice the plots for different classes in Task 4). Even for classes introduced as part of the same task (class 47, Nausea and class 49, Punching both introduced in Task 2), their ordered prompt selection is unique and consistent even after updating the model on new data in subsequent continual sessions.

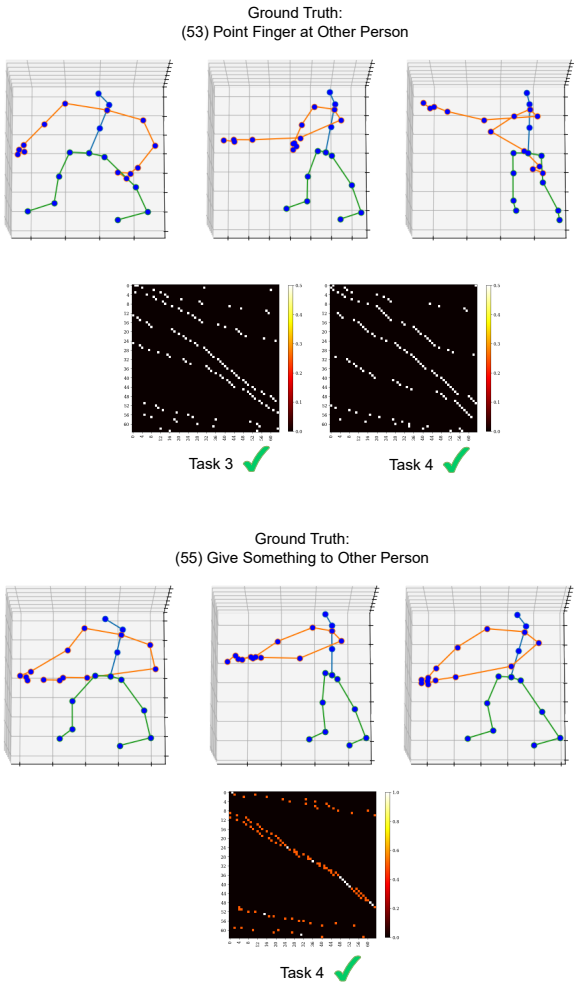


Fig. 10: Instance-Level Ordered Prompt Selection using POET: Our proposed method POET is an input instance-based prompt tuning approach for FSCIL, as the prompts are selected conditioned on each input instance itself. Hence, here we study instance-level prediction on the test set. The sample of class *Point Finger, class ID 53* is evaluated after $\mathcal{US}^{(3)}$ and $\mathcal{US}^{(4)}$ as the class was added to the model in $\mathcal{US}^{(3)}$. The sample of class *Give Something, class ID 55* is continually learnt and evaluated after $\mathcal{US}^{(4)}$. We point out the unique ordered key index sequence for the 2 instances, which could have been easily confused by the model due to their semantic similarity. The ordering matrix for *Point Finger* remains consistent across tasks, even after adding 5 new classes in $\mathcal{US}^{(4)}$.

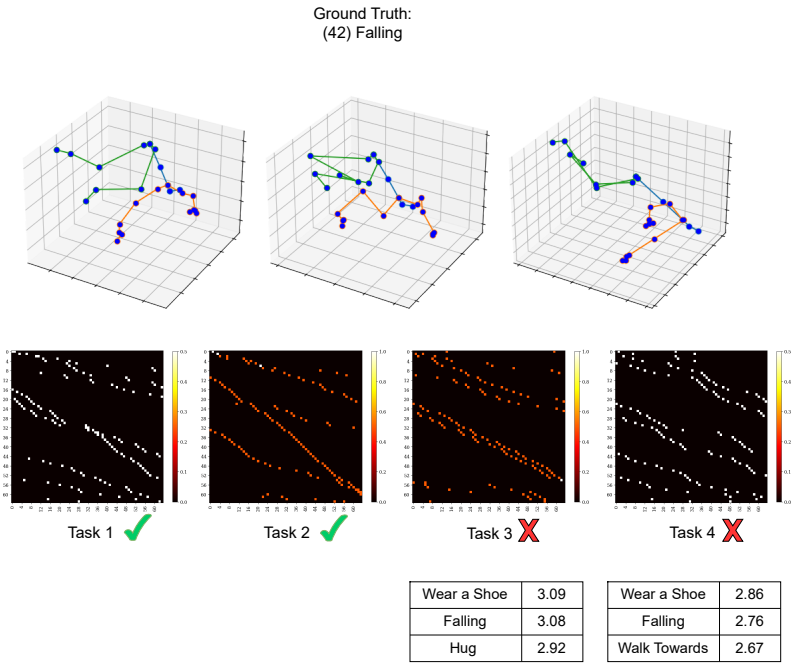


Fig. 11: Instance-Level Ordered Prompt Selection using POET: We also show a failure case of our proposed approach. After learning the class *Falling* in $\mathcal{US}^{(1)}$, we evaluate it after every new continual task. Even though it correctly predicts a test set instance in $\mathcal{US}^{(1)}$ and $\mathcal{US}^{(2)}$, it tends to get confused by the class *Wearing a Shoe* at $\mathcal{US}^{(3)}$ and $\mathcal{US}^{(4)}$. Notice, this coincides with a disruption in the ordering statistics.