Coherent Temporal Synthesis for Incremental Action Segmentation

Guodong Ding, Hans Golong and Angela Yao
National University of Singapore
{dinggd, hgolong, ayao}@comp.nus.edu.sg

Abstract

Data replay is a successful incremental learning technique for images. It prevents catastrophic forgetting by keeping a reservoir of previous data, original or synthesized, to ensure the model retains past knowledge while adapting to novel concepts. However, its application in the video domain is rudimentary, as it simply stores frame exemplars for action recognition. This paper presents the first exploration of video data replay techniques for incremental action segmentation, focusing on action temporal modeling. We propose a Temporally Coherent Action (TCA) model, which represents actions using a generative model instead of storing individual frames. The integration of a conditioning variable that captures temporal coherence allows our model to understand the evolution of action features over time. Therefore, action segments generated by TCA for replay are diverse and temporally coherent. In a 10-task incremental setup on the Breakfast dataset, our approach achieves significant increases in accuracy for up to 22% compared to the baselines.

1. Introduction

Ensuring that intelligent systems can continually adapt and accumulate knowledge in our rapidly evolving world is essential. This concept is embodied in incremental learning [8]. A key challenge is to acquire new knowledge gradually without catastrophic forgetting [15] of previously learned information. There have been numerous efforts to tackle the problem in the machine learning community, including data replay [39], regularization techniques [21], and knowledge distillation [32].

Data replay is an effective and commonly used technique in image incremental learning [2,4,43,52]. Data replay mitigates catastrophic forgetting by re-exposing the model to previously encountered data samples. Previous works either retain a subset of the original training samples [18,34,39] or learn a generative model on previous training data to later generate surrogate training samples [17, 28, 43]. A recent shift of incremental learning from static images to dynamic videos has brought about the application of data replay into the video domain [3, 36, 50]. However, this transition has been limited to direct implementations, with less emphasis on videos' unique temporal properties.

Action recognition [23], as the hallmark task of video understanding, was the initial focus for incremental learning for videos [3, 36, 37, 50]. Many works create replay data by storing frames for each video clip. Storing frame-exemplars for videos requires a balance between *dense temporal sampling* and *memory diversity* [3, 50]. It is a trade-off between storing fewer complete videos or more incomplete ones within a given memory budget. Because existing recognition models accept sparse frames as input, like 8 or 16 frames for TSN [51] and TSM [33], keeping an exemplar set for each action clip does not overly strain the memory allocation. Also, there is a strong static bias towards scene context [30] in standard action recognition datasets like Kinetics [7] and UCF101 [46]. It is, therefore, possible to achieve comparable performance storing either entire segments, a few frames [50], or even a single frame [3], all without the need for explicit temporal modeling. As a result, most techniques save exemplar frames and mirror the practices from the image domain while giving less attention to developing video-specific data replay techniques.

We advocate exploring data replay strategies in incremental video understanding with *temporal action segmentation* (TAS) [9]. TAS is akin to semantic image segmentation [35] but from a one-dimensional temporal basis. Instead of 2D pixel-wise semantic labels, TAS assigns frame-level action labels. For instance, given a procedural video of "*making coffee*", a TAS model classifies each frame as a step such as '*take cup*', '*pour coffee*', '*pour milk*', *etc.* In TAS, a procedural task is usually called an "*activity*" and the steps '*actions*'. The necessity of temporal modeling in TAS makes it more suitable for investigating video-specific replay strategies. First, static bias is less prominent in TAS since different actions within a sequence often share a common background, emphasising the importance of temporal modeling for video data replay. Second, TAS operates with full-resolution frame-by-frame inputs and produces outputs at the same level of temporal granularity. The extended temporal span requires more capacity in the replay memory compared to shorter, trimmed action clips. Storing frame exemplars in proportion to the video's duration, even downsampled, imposes a significant memory burden. Using sparse frames to represent actions also results in a loss of temporal coherence in the natural progression of actions.

We present a novel temporally coherent action model for video data replay in incremental action segmentation. Contrary to the conventional frame-exemplar storing approaches, we use a generative model to represent actions. This is motivated by the model's capacity to learn efficient data representations with a fixed model size while producing diverse outputs of arbitrary lengths. The replay data generation is top-down. First, a replay video is defined by its sequential structure, including action sequences and segment durations. Subsequently, the generative model generates features for each action segment. Finally, these generated segments are concatenated to form a complete replay video. Unlike generative approaches for image tasks [17,28,43], our model incorporates a conditioning variable to account for the unique temporal coherence of videos. The coherence variable, defined as the relative progression within an action, assists the model in capturing the evolution of action features over time.

Contributions. Our contributions are summarized as follows: **(1)** To the best of our knowledge, we are the first to introduce the incremental action segmentation task, working with procedural videos. This is a natural fit for the development of intelligent assistants which learn complex tasks and activities in the real world in an incremental manner. **(2)** We propose to model actions with generative models and use the models to generate diverse replay data. The proposed generative data replay bypasses the trade-off problem in frame-exemplar storing approaches. **(3)** We introduce a temporal coherence variable to help the generative model learn action feature evolution and produce temporally coherent action segments in replay video generation. **(4)** Experiments on two procedural benchmarks show that our approach can effectively mitigate catastrophic forgetting for incrementally learned action segmentation models.

2. Related Works

Incremental Learning. Incremental learning involves algorithms adapting to new data without forgetting prior knowledge. Popular approaches in the image domain include data replay [39], regularization techniques [21], and knowledge distillation [32]. Data replay methods in the image domain are categorized into direct replay, using the original exemplar set for rehearsal [18, 34, 39], and generative replay, which models sample distribution to generate instances [17, 28, 43]. Limited attention has been given to the video domain [3, 50], and most works rely on direct replay of frame-wise exemplars while neglecting temporal aspects. This work proposes leveraging generative models, focusing on maintaining temporal coherence for video data replay.

Temporal Action Segmentation. There have been many existing approaches proposed for the temporal action segmentation task. Fully supervised approaches rely on the dense annotation of the video frames [12, 53]. In a semi-supervised setting [10, 44], only a subset of videos requires dense labels, while the remaining videos are unlabeled. Weaker forms of supervision include action transcripts [25], action sets [14, 29, 40], timestamps [31, 38] and activity labels [11]. Some cases [26, 41, 42] work without any action labels in an unsupervised setup. Despite the ongoing efforts in TAS with diverse forms of supervision, incremental learning has not been explored. Our work is the first to study incremental action segmentation, emphasizing video replay techniques.

3. Preliminaries

Temporal Action Segmentation (TAS) divides untrimmed video sequences into temporal segments and associates each segment with a predefined action label [9]. Given a video $x^{1:T} = \{x^1, ..., x^T\}$ of T frames long, a model $\mathcal{M}$ segments x into N contiguous and non-overlapping actions:

$$\mathbf{s}_{1:N} = (s_1, s_2, ..., s_N), \quad \text{where } s_n = (a_n, t_n, \ell_n), \quad (1)$$

$$\text{s.t.} \qquad t_{n+1} = t_n + \ell_n,$$

s_n denotes a temporal segment in the video of length of ℓ_n, with action class label $a_n \in \mathcal{A}$ from A predefined categories. t_n denotes the starting timestamp of segment s_n and adheres to a precise temporal sequence from the preceding segment. In practice, most existing works [12, 27, 45, 53] design $\mathcal{M}$ to classify actions on a per-frame basis, *i.e*.,

$$y^{1:T} = (y^1, y^2, ..., y^T), \quad (2)$$

where $y^t \in \mathcal{A}$ is the frame-wise action label at time t.

Common architectures for the segmentation model $\mathcal{M}$ include convolution-based MSTCN [12] and transformer-based ASFormer [53]. Due to the memory constraints, x^t is typically provided as pre-computed features such as I3D [7] rather than raw RGB inputs. The segmentation model $\mathcal{M}$ is trained with a frame-wise cross-entropy loss:

$$\mathcal{L}_{\text{cls}}(x, y) = \frac{1}{T}\sum_t - \log(p^t(y^t)), \quad (3)$$

where $p^t \in \mathbb{R}^A$ is the estimated action probabilty for the frame x^t. In addition, a smoothing loss is imposed to ensure smooth transitions between consecutive frames:

$$\mathcal{L}_{\text{sm}}(x, y) = \frac{1}{TA}\sum_{t,a} \tilde{\Delta}^2_{t,a}, \quad \tilde{\Delta}_{t,a} = \begin{cases} \Delta_{t,a} : \Delta_{t,a} \leq \tau \\ \tau \quad\ : \text{otherwise} \end{cases}, \quad (4)$$

$$\Delta_{t,a} = \left|\log p^t(a) - \log p^{t-1}(a)\right|.$$

τ is conventionally set to 4, as per [12]. The full training loss is written as the combination of the above two, balanced by hyperparameter λ:

$$\mathcal{L}_{\text{tas}} = \mathcal{L}_{\text{cls}} + \lambda \cdot \mathcal{L}_{\text{sm}}. \tag{5}$$

Class Incremental Learning (CIL) introduces new classes to model $\mathcal{M}$ over time [39]. Consider a series of B tasks, each task $\mathcal{T}_b$ represents a distinct incremental step, comprising a set of n_b training instances denoted as $\{(x_i, y_i)\}_{i=1}^{n_b}$. Here, x_i represents an instance associated with class $y_i \in \mathbf{Y}_b$, where $\mathbf{Y}_b$ is the class set for task b. Importantly, access to the training data in $\mathcal{T}_b$ is restricted to training task b only, and there is no longer access to data from tasks prior to b. The developed model must acquire knowledge from the current task while preserving knowledge from the past tasks. The performance of an incremental learning model is evaluated over all seen classes, *i.e.*, $\mathcal{Y}_B = \mathbf{Y}_1 \cup ... \mathbf{Y}_B$. Standard CIL assumes non-overlapping classes in different tasks ($\mathbf{Y}_b \cap \mathbf{Y}'_b = \emptyset$ for $b \neq b'$), but when class overlap occurs, the task referred to as blurry class-incremental learning (Blurry CIL) [4, 5, 22].

Data replay is a commonly adopted approach for incremental learning by revisiting former exemplars. The exemplar set, also known as the replay buffer, is an extra collection of instances from the previous tasks $\hat{\mathcal{T}}_{1:b} = \{(\mathbf{x}_{b'}, \mathbf{y}_{b'})\}_{b'=1}^{b-1}, y_{b'} \in \mathcal{Y}_{b-1}$. The exemplars can either be specified training samples [39], constructed prototypes [19] or through generation [43, 52]. The model can then utilize $\mathcal{T}_b \cup \hat{\mathcal{T}}_{1:b}$ for update while attaining previous knowledge.

4. Incremental Temporal Action Segmentation

Incremental temporal action segmentation (iTAS) continuously updates an action segmentation model when encountering new action classes over time. Unlike existing CIL task that assumes no interclass dependencies, iTAS works with procedural videos where the same activity often share a common set of actions, making it more intuitive to treat each procedural activity as an incremental task b and the actions within each activity as class labels $\mathbf{Y}_b$. The segmentation model $\mathcal{M}$ learns to segment videos from various incremental activities. This work introduces a novel generative data replay approach for iTAS featuring a temporally coherent action model.

4.1. Temporally Coherent Action Modeling

Generative models, such as Variational Autoencoders (VAE) [20] and Generative Adversarial Networks [16] are powerful tools for learning efficient representations of data with a fixed model size. Additionally, they are easy to extend to a conditional form to generate conditioned outputs.

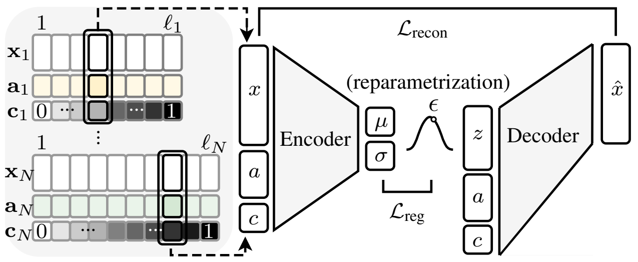


Figure 1. Temporally Coherent Action (TCA) model. The input to the encoder is the concatenation of the frame feature x, action label a and coherence variable c. The decoder samples a latent variable with reparametrization: $z = \mu + \sigma \odot \epsilon, \epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{1})$, and outputs $\hat{x}$ as the reconstruction of the original feature.

These properties make them a good fit for the purpose of generating diverse action segments for data replay.

In this work, we employ a conditional VAE to model the actions. Other generative models are also feasible, but conditional VAEs offer a good tradeoff between model size, expressiveness, and data efficiency for learning. In a conditional VAE model, an encoder maps an input frame feature x and the conditioning action class label a to a probability distribution $q_\phi(z|x, a)$ over the latent space, where z is the latent variable. The decoder maps a sample from latent space z and the conditioning information back to obtain a reconstruction of the input $\hat{x} = p_\theta(x|z, a)$. ϕ, θ are learnable network parameters of the decoder and encoder, respectively. The overall loss function is a combination of the reconstruction term and the KL divergence regularization term, written as:

$$\mathcal{L}_{\text{cVAE}} = \underbrace{\mathbb{E}_z \log p_\theta(x|z, a)}_{\text{reconstruction}} - \underbrace{\mathrm{D}_{\text{KL}}(q_\phi(z|x, a)||p(z))}_{\text{regularization}}. \tag{6}$$

The second term regularizes the latent space by a prior $p(z)$ on the approximate posterior distribution; typically, the prior is a Gaussian distribution.

Temporal Coherence Modeling. The above action model only captures the diversity of action segments but ignores any temporal coherence since the action frames are modeled independently. To that end, we introduce a coherence variable to model the temporal transition between feature frames within an action segment. The coherence is defined as the relative temporal progression of a frame within the action. For the i-th frame in an action segment of duration ℓ, the coherence variable c_i is defined as:

$$c_i = (i-1)/(\ell-1), \text{ and } c_i \in [0, 1]. \tag{7}$$

Incorporating the relative progression as the coherence condition helps the model build up knowledge of the feature continuity following the progression of actions. In this paper, we follow [6] and assume the action progression to be

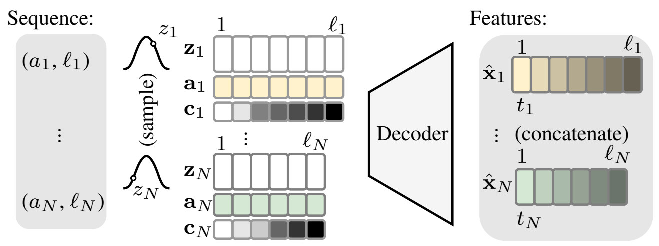


Figure 2. Replay Data Generation. In each segment, the action variable a remains consistent, and a common latent variable $z \sim \mathcal{N}(\mathbf{0}, \mathbf{1})$ is sampled for all frames. The resulting features $\hat{\mathbf{x}}$ by the decoder exhibit continuity due to the gradual change in the coherence variable c. The segments are then concatenated following their starting timestamp t to form a complete video.

linear. Adding upon Eq. (6), the training loss for our Temporally Coherent Action (TCA) model is written as:

$$\mathcal{L}_{\text{TCA}} = \underbrace{\mathbb{E}_z \log p_\theta(x|z, a, c)}_{\mathcal{L}_{\text{recon}}} - \underbrace{\mathrm{D}_{\text{KL}}(q_\phi(z|x, a, c)||p(z))}_{\mathcal{L}_{\text{reg}}}. \tag{8}$$

Fig. 1 illustrates the TCA model. For each frame in a segment, the encoder $\mathcal{E}(x, a, c) = q_\phi(z|x, a, c)$ takes in the feature x, one-hot action label a, and its coherence variable c, while the decoder $\mathcal{D}(z, a, c) = p_\theta(x|z, a, c)$ samples a latent variable z and uses the same a, c for reconstruction.

4.2. Replay Data Generation

Our replay data generation is top-down; it first samples a sequential structure before generating the action segments.

Sequential Structure Sampling. As described by the segment-level interpretation provided by Eq. (1), a procedural video can be summarized as an ordered sequence of actions, each with varying durations. Such a high-level structure helps guide the data generation process. The (symbolic) sequences require negligible storage, so we keep all the sequences from the training data and directly sample from the set during generation. Specifically, we establish a candidate pool to store all action sequence order and their durations $\mathbf{S}_b = \{\mathbf{s}_i\}_{i=1}^{n_b}, \mathbf{s}_i \in \mathcal{T}_b$. During the data generation stage, we employ a uniform sampling strategy on $\mathbf{S}_b$:

$$\hat{\mathbf{s}}_b \sim \text{Uniform}(\mathbf{S}_b). \tag{9}$$

Action Segment Generation. The sample sequential structure $\hat{\mathbf{s}} = \{(a_n, t_n, \ell_n)\}_{n=1}^{\hat{N}}$ is then utilized in the segment generation process, as illustrated in Fig. 2. For each sampled (a_n, t_n, ℓ_n), our TCA model generates the frame-wise feature for the segment using:

$$\hat{x}_i = p_\theta(x|z_n, a_n, c_i), \text{ and } i \in [1, ..., \ell]. \tag{10}$$

Here, we fix z_n across all frames for the given action a_n and vary the coherence variable c_i, following Eq. (7). This

```
Algorithm 1 Incremental Temporal Action Segmentation
Input: Task video data $\{\mathcal{T}_1, ..., \mathcal{T}_B\}$, replay size $M$
Output: Segmentation model $\mathcal{M}$
 1: Train $\mathcal{M}$ with data $\mathcal{T}_1$;                              ▷ Eq. (5)
 2: Train $(\mathcal{E}_1, \mathcal{D}_1)$ with data $\mathcal{T}_1$;           ▷ Eq. (8)
 3: for $b = 2, ..., B$ do                                   ▷ Incremental Learning
 4:     Initialize replay data $\hat{\mathcal{T}}_{1:b} = \emptyset$;
 5:     for $b' = 1, ..., b-1$ do                        ▷ Replay Data Generation
 6:         Get $\hat{\mathbf{s}}_{b'}$ from $\mathbf{S}_{b'}$ for $M/(b-1)$ times;   ▷ Eq. (9)
 7:         Generate $\hat{v}_{b'}$ for each $\hat{\mathbf{s}}_{b'}$ with $\mathcal{D}_{b'}$;   ▷ Eq. (11)
 8:         $\hat{\mathcal{T}}_{1:b} \leftarrow \hat{\mathcal{T}}_{1:b} \cup \{\hat{\mathbf{v}}_{b'}\}_{M/(b-1)}$;
 9:     end for
10:     Train $\mathcal{M}$ with data $\mathcal{T}_b \cup \hat{\mathcal{T}}_{1:b}$;          ▷ Eq. (12)
11:     Train $(\mathcal{E}_b, \mathcal{D}_b)$ with data $\mathcal{T}_b$;              ▷ Eq. (8)
12: end for
```

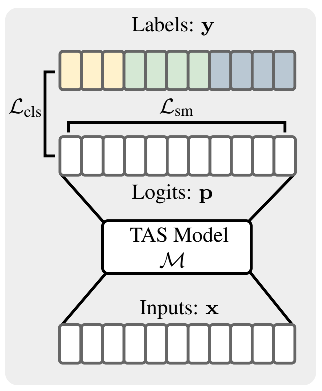
ensures temporal coherence in feature transition within the segment. The process is repeated for every segment, and these generated segment features $\hat{\mathbf{x}}_n$ are concatenated according to their timestamps t_n to compose the procedural video:

$$\hat{v} = \text{concat}(\hat{\mathbf{x}}_1, ..., \hat{\mathbf{x}}_{\hat{N}}). \tag{11}$$

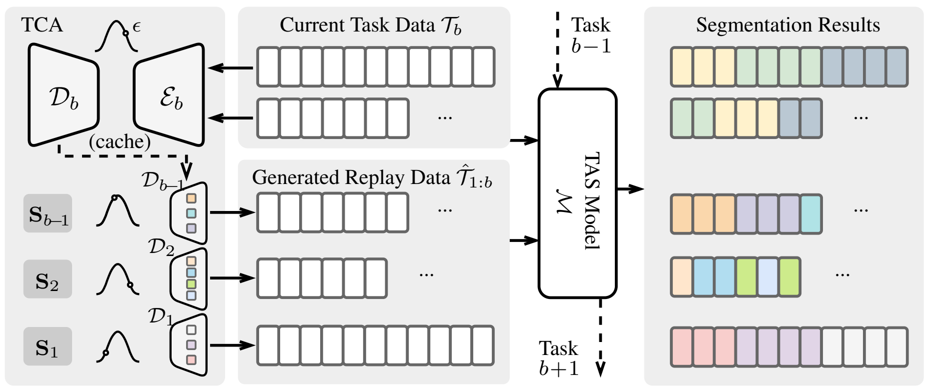
Discussion. An intuitive interpretation of the latent and conditioning variables in Eq. (10) can be as follows: z governs the overall scene context, a regulates the action semantics, and c manages the temporal progression of the action features. Since the action label a is predefined for each segment, it is kept consistent. However, multiple generation options for the action segments can be achieved by manipulating the latent variable z and the conditioning coherence variable c. For instance, we can generate a *static* segment with a constant latent variable z and a constant coherence c spanning the entire segment. On the other hand, by fixing c, we can generate a dynamic yet *random* segment with z_i independently sampled for each frame. We show in Sec. 5.4 that both diversity and coherence are essential in video data replay. We ensure temporal coherence within segments but not across action transitions because boundary frames, as indicated in [10, 47], are ambiguous and may not be conducive to learning the segmentation model.

4.3. Incremental Training

We commence by training the segmentation model $\mathcal{M}$ (Fig. 3(a)) and our TCA model $(\mathcal{E}_1, \mathcal{D}_1)$ with the initial task data $\mathcal{T}_1$. As new task data $\mathcal{T}_b$ becomes available, we create replay video data $\hat{v}$ for all previous tasks $[1 : b)$ utilizing their corresponding TCA decoders $\mathcal{D}$. This process yields a total of M videos denoted as $\hat{\mathcal{T}}_{1:b}$. To train the segmentation model $\mathcal{M}$, we combine this generated data with the real data from $\mathcal{T}_b$ to serve as the training set, as shown in Fig. 3(b). Through this, the segmentation model can revisit the previous tasks to mitigate catastrophic forgetting while learning



(a) Standard TAS



(b) Incremental TAS with TCA Replay

Figure 3. Learning schemes for temporal action segmentation. (a) Standard TAS and (b) Incremental TAS with TCA at task b.

new tasks. The incremental learning objective at task b is as follows:

$$\mathcal{L}^b_{\text{iltas}} = \underbrace{\mathcal{L}_{\text{cls}}(x,y)}_{(x,y)\in\mathcal{T}_b\cup\hat{\mathcal{T}}_{1:b}} + \lambda \cdot \underbrace{\mathcal{L}_{\text{sm}}(x,y)}_{(x,y)\in\mathcal{T}_b\cup\hat{\mathcal{T}}_{1:b}}, \tag{12}$$

we set $\lambda = 0.15$, following [12]. Concurrently, as iterating through the above steps, we train our TCA model $(\mathcal{E}_b, \mathcal{D}_b)$ every time new video data from task b arrives. The overall incremental learning procedure is summarized in Algorithm 1.

5. Experiments

5.1. Datasets and Evaluation

Datasets. We adapt Breakfast [24] and YouTube Instructional [1] datasets to incremental learning. Note other TAS datasets such as GTEA [13] and 50Salads [48] are unsuitable because most of their videos share a common set of actions, making it challenging to implement a meaningful number of incremental tasks without actions overlapping. **Breakfast** [24] dataset comprises 1,712 undirected breakfast preparation videos. There are 10 activities and a total of 48 action classes; each video features 5 to 14 actions. Each activity comprises around 150 videos for training and 20–30 for testing. We use the I3D [7] feature representations and evaluate with the standard splits. **YouTube Instructional (YTI)** dataset [1] features 150 instructional videos of 5 activities (30 each). There are a total of 46 actions, each activity featuring 6 to 13 actions. We use 80% of the videos in each activity for training and reserve the remaining for evaluation. We use the same set of feature representations as [26, 42].

Incremental Learning. We partition the Breakfast and YTI datasets based on activities, each activity as a separate task, and train models incrementally. We experiment with both disjoint and blurry incremental settings on Breakfast. Disjoint tasks consider shared actions different, *e.g.*, *'take plate'* in *"friedegg"* is a different class from *'take plate'* in *"sandwich"*. The blurry setting allows overlapping and assigns common actions across tasks the same class label.

Evaluation Measures. TAS is evaluated by a frame-wise accuracy (Acc), segment-wise edit score (Edit), and F1 score with varying overlap thresholds of 10%, 25%, and 50%. We adopt these standard measures and apply them to the incremental setup. We denote the Acc on task b' after the b-th task ($b' < b$) as $\text{Acc}^{b'}_b$, and use the last stage accuracy averaged over all asks as the final metric to measure the overall performance[1], *i.e.*,

$$\text{Acc} = \sum_{b=1}^{B} \text{Acc}^b_B. \tag{13}$$

The Edit and F1 scores are similarly defined.

5.2. Implementations

TCA Model. Our TCA model comprises an encoder and decoder; each implemented as a two-layer MLP with a 256-D latent space. Based on their dataset size, we train TCA for 2,500 epochs with Breakfast and 250 epochs for YTI. The learning rate is set to be $1e^{-3}$ and $1e^{-5}$, respectively.

TAS Backbones. We experiment with the temporal convolutional network MSTCN [12] and the transformer-based ASFormer [53] for our backbone model $\mathcal{M}$. MSTCN is trained with a learning rate of $5e^{-4}$ for 50 epochs for each task; for ASFormer, it is $1e^{-4}$ for 30 epochs.

Baselines. We first use a naive (**'Finetune'**) baseline that progressively finetunes with only training data in the task sequence without data replay. Additionally, we follow the boring baseline from action recognition [3] and store a mean

[1]We calculate the average performance over all tasks due to the imbalance in frame numbers across each task.

# Tasks		MSTCN [12]					ASFormer [53]				
		Acc	Edit	F1 @ {10, 25, 50}			Acc	Edit	F1 @ {10, 25, 50}		
						Breakfast					
10	Finetune	7.4	7.2	7.5	7.0	5.4	9.9	9.8	10.3	9.4	7.5
	Exemplar [3]	16.1	13.3	13.8	12.5	9.5	12.4	11.2	11.7	10.7	8.5
	Ours	**29.4**	**25.9**	**26.3**	**23.5**	**17.7**	**34.2**	**32.4**	**33.1**	**30.1**	**23.4**
	Original	43.1	41.1	41.2	37.6	29.5	48.1	45.2	45.9	42.4	34.2
5	Finetune	15.4	15.8	16.6	15.8	12.7	15.7	16.1	16.9	15.8	13.2
	Exemplar [3]	32.5	28.9	30.8	28.5	22.9	29.5	27.5	28.7	26.7	22.0
	Ours	**54.5**	**49.4**	**51.1**	**46.9**	**37.7**	**57.2**	**56.8**	**58.3**	**54.0**	**43.6**
	Original	60.4	59.1	60.3	56.1	46.0	65.1	64.2	65.6	61.5	51.0
						YouTube Instructional					
5	Finetune	13.6	2.8	3.6	2.7	0.6	13.9	11.5	11.1	9.8	6.3
	Exemplar [3]	**30.8**	19.7	19.8	16.0	9.3	22.1	18.9	17.7	15.3	10.0
	Ours	30.2	**25.0**	**21.9**	**18.5**	**11.1**	**25.2**	**20.9**	**20.1**	**17.5**	**11.4**
	Original	55.9	39.4	38.1	32.2	19.1	59.2	51.1	45.4	39.1	25.5

Table 1. Performance comparison on Breakfast and YouTube Instructional. Our approach consistently surpasses the baselines, with both MSTCN and ASFormer backbones. (Results are averaged over five random seeds, refer to Supplementary Materials for variations.)

frame for each action segment per sequence, which we call **'Exemplar'**. Such a segment is static, as the frame-wise sample is simply replicated to inflate the segment in time and used for data replay. Finally, we consider an upper-bound (**'Original'**) data replay baseline that uses the original sequence features.

Replay Size. For all experiments, we constrain the maximum number of videos for replay to be 60[2], which is equally divided by the number of seen tasks. Given that our sequence is sampled from seen videos, the total frame number in the generated set is bounded by the 60× longest video. Our ablation study (see Sec. 5.4) shows that the performance can be further improved with a larger replay size.

5.3. Effectiveness

We compare the performances on two datasets in Tab. 1. Finetune achieves the lowest performance as it simply concentrates on learning new concepts without revisiting previous tasks. In the 10-task setup on Breakfast, static segments with stored exemplar features mitigate forgetting, gaining an accuracy increase from 7.4% to 16.1% with MSTCN. Our approach achieves a more substantial boost, reaching 29.4%. The improvement is consistent across both backbones. Despite this improvement, there remains a 13.1% gap compared to using original features from previous tasks. There are similar performance differences across the approaches in the 5-task incremental step. The 5-task incremental setup combines every two activities as a single task and has fewer training steps, improving performance across all approaches. This suggests that the challenge of learning increases with the number of tasks, leading to increased forgetting by the model. The performance improvement for the YTI dataset is less significant compared to Breakfast; both the Exemplar [3] and our method achieve very similar performances. We hypothesize that this phenomenon results from the overall reduced segment diversity in the YTI dataset due to its comparatively smaller size. Nevertheless, the rise in the segmental metrics suggests that temporally evolving action segment features, as opposed to static ones, can alleviate over-segmentation. When comparing the backbones, ASFormer [53] is prone to overfit to static data compared to MSTCN [12]. The performance on Breakfast demonstrates that ASFormer outperforms MSTCN across all approaches except in Exemplar, where static segments are employed for replay.

Blurry CIL. Our evaluation using the blurry setup on the Breakfast dataset is presented in Tab. 2. We consistently observe performance improvements, with all performance metrics surpassing the standard setup (Tab. 1) by 10 points. This improvement is likely attributed to the efficient updating of action classifiers from previous tasks with the current task samples due to shared actions and blurred task boundaries.

5.4. Ablation Study

Feature Diversity and Temporal Coherence. In our evaluation of the impact of feature diversity and temporal coherence on performance (see Tab. 3), we assess three key factors: the diversity of actions at the segment level, the diversity of frames within a segment, and the temporal coherence of a segment. Both with static segments, $\text{Ours}_{\text{static}}$ outperforms the Exemplar by approximately 10% in accuracy (Rows 1 *vs.* 3), which showcases TCA's ability to gen-

[2]We choose the value of 60 to maintain a similar exemplar-to-training ratio (1:25) per iCIFAR-100 used in [39].

	MSTCN [12]					ASFormer [53]				
	Acc	Edit	F1 @ {10, 25, 50}			Acc	Edit	F1 @ {10, 25, 50}		
Finetune	15.8	29.7	30.0	26.0	19.1	15.5	30.2	30.6	26.4	19.6
Exemplar [3]	25.0	34.0	34.9	30.2	22.3	24.4	35.6	37.2	32.9	24.9
Ours	**38.5**	**43.3**	**44.9**	**39.5**	**29.7**	**44.2**	**51.2**	**53.0**	**47.6**	**36.8**
Original	48.5	53.4	55.1	49.4	37.9	53.5	57.6	59.9	54.5	43.2

Table 2. Performance comparsion on Breakfast with the blurry task boundary.

	SD	FD	TC	Acc	Edit	F1 @ {10, 25, 50}		
Exemplar	✓	✗	✗	27.8	35.6	36.1	31.7	24.3
Ours$_{\text{random}}$	✓	✓	✗	32.9	38.9	40.0	35.6	27.2
Ours$_{\text{static}}$	✓	✗	✗	37.9	42.9	43.8	38.9	29.0
Ours	✓	✓	✓	**41.8**	**45.0**	**47.0**	**41.5**	**32.0**

Table 3. Ablation study on the components in TCA. 'SD' denotes the diveristy between segments of the same action, 'FD' denotes the diversity between the frames within the same segment, and 'TC' denotes the temporal coherence of the segments. Optimal results are obtained when considering both diversity and temporal coherence. Gray row indicates the final setup used in this paper.

M	Acc	Edit	F1 @ {10, 25, 50}		
30	34.0	39.6	41.0	34.8	24.7
60	35.4	41.2	42.3	36.0	25.6
90	36.2	**42.3**	43.9	**37.3**	**26.8**
120	**38.0**	**42.3**	**44.0**	37.1	26.2

Table 4. Different replay sizes on Breakfast. Revisiting more replay videos can help the model mitigate forgetting.

	$\mathcal{T}(\%)$	Acc	Edit	F1 @ {10, 25, 50}		
Exemplar	-	22.6	34.8	36.0	32.4	25.2
Ours	25	41.7	43.2	46.1	40.9	31.5
	50	42.1	43.3	45.1	40.5	31.5
	75	45.3	45.9	47.8	**43.7**	**34.7**
	100	**47.4**	**46.9**	**48.2**	42.8	33.4

Table 5. Different ratios for TCA training on Breakfast. The final performance is correlated with the quantity of data used in TCA training. A higher volume of training data leads to improved final perfromance.

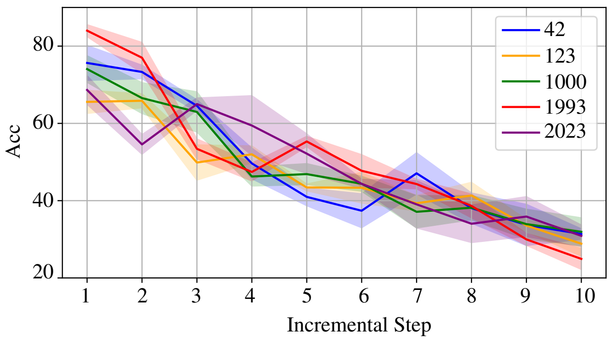


Figure 4. Performance following each incremental step with the varied task sequence. The mean and standard deviation across four splits are represented by solid lines and shaded areas, respectively. Each unique task sequence, determined by a random seed, is distinguished by a different color.

erate diverse actions and enhance segment-level diversity. However, Ours$_{\text{random}}$, which maximizes feature diversity but lacks temporal constraints, performs less effectively than Ours$_{\text{static}}$, highlighting the detrimental impact the absence of temporal constraints can cause for the TAS task. In the final row, our approach, which coordinates feature diversity and temporal coherence, achieves the highest performance.

Replay Size. We examine four different replay sizes: $M = 30, 60, 90, 120$. As depicted in Tab. 4, performance is observed to be correlated with the replay size M as anticipated. Employing only 30 video sequences as replay data results in a marginal decline (34.0%) compared to 60 (35.4%). An increase in the replay size M corresponds to incremental improvements in both frame-wise and segmental metrics, and the best performance is achieved with the replay size being set to 120.

TCA Training Data. The results of training TCA with varying proportions of task data $\mathcal{T}$ are presented in Tab. 5. A consistent trend indicates that increased access to task data during TCA training brings greater overall performance improvements and less forgetting. Training TCA with the entire task data yields the overall best performance, with a significant 6% decrease in performance observed when only a quarter of the data is utilized. Notably, our approach outperforms the Exemplar significantly, even when trained with only 25% of task data.

Task Sequence. We compare five distinct task sequence arrangements within a 10-task incremental setup on Breakfast and report the results in Fig. 4. Although there is a consistent decrease in accuracy scores, the ultimate performance is subject to variation depending on the seed used for task sequence determination, showing differences of up to 7% points. This suggests that the forgetting in incremental learning is related to the order of these learning tasks. Moreover, the disparity could be exacerbated by the uneven

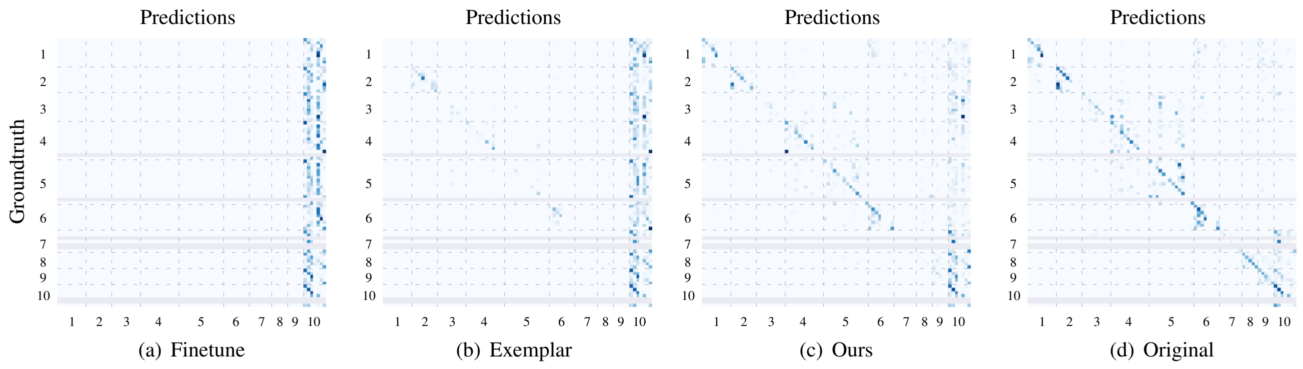


Figure 5. Comparsion of confusion matrices of different approaches on Breakfast in a 10-task setup. The task sequence is given as follows: 1 - "*sandwich*", 2 - "*juice*", 3 - "*friedegg*", 4 - "*scrambledegg*", 5 - "*pancake*", 6 - "*salat*", 7 - "*tea*", 8 - "*milk*", 9 - "*cereal*", 10 - "*coffee*". The dashed lines indicate task boundaries, while gray rows denote the lack of instances for that action in the test data. Our approach (c) shows the closest resemblance to data replay using original features (d).

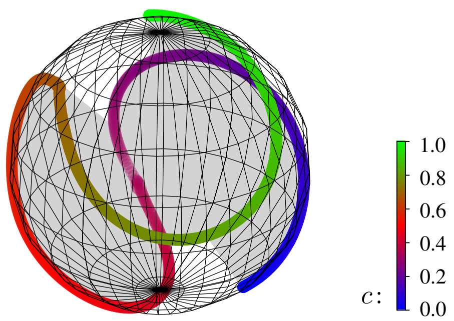


Figure 6. T-SNE visualization of a generated action segment given the action '*pour milk*' from activity "*cereal*". Every point in the sphere corresponds to a frame feature, and the continuity of these feature points demonstrates temporal coherence.

distribution of frames across tasks, which is commonly observed in procedural videos.

5.5. Visualization

Confusion Matrix. In Fig. 5, we plot the confusion matrices generated by different approaches. Our method demonstrates superior performance compared to Finetune and Exemplar, particularly in the case of longer activities such as "*sandwich*", "*scrambledegg*", "*pancake*" and "*salat*". Furthermore, similar to Original, our approach exhibits increased confusion between semantically similar activities, such as "*scrambledegg*" and "*pancake*", both involving cooking with a pan.

Temporal Coherence. To assess the temporal coherence, We employ T-SNE [49] to visualize an exemplary action segment generated by our TCA model. Specifically, we employ the TCA model trained on "*cereal*" data to generate an action segment of '*pour milk*'. This segment comprises 1,000 frames, each sharing a common action label a and latent variable z, with an evenly strided c. As shown in Fig. 6, the features of the action segment exhibit continuous properties in the feature space, corresponding to the gradual increase in the coherence variable c.

5.6. Limitations

This work presupposes the presence of dense labels for all videos within each incremental task, and our TCA model relies on such dense labels for effective learning. However, such an annotation process can be expensive in a real-world scenario. Additionally, while we can demonstrate the temporal coherence of the features generated by our TCA model, the qualitative evaluation of the features remains challenging because their existence in a feature space makes them difficult to decode into a human-readable format.

6. Conclusion

This paper proposes a temporally coherent action model for incremental procedural video understanding from the perspective of video data replay. Departing from the conventional frame-exemplar data replay prevalent in action recognition, our approach adopts a generative model. In addition, we facilitate the modeling of temporal coherence in the actions by introducing a conditioning variable to the generative model. Our design guarantees both feature diversity and temporal coherence in the replay data, resulting in a significant performance improvement on two action segmentation benchmarks compared to the baselines.

Acknowledgement

This research is supported by the National Research Foundation, Singapore under its NRF Fellowship for AI (NRF-NRFFAI1-2019-0001). Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not reflect the views of National Research Foundation, Singapore.

References

[1] Jean-Baptiste Alayrac, Piotr Bojanowski, Nishant Agrawal, Josef Sivic, Ivan Laptev, and Simon Lacoste-Julien. Unsupervised learning from narrated instruction videos. In *CVPR*, 2016. 5

[2] Rahaf Aljundi, Min Lin, Baptiste Goujaud, and Yoshua Bengio. Gradient based sample selection for online continual learning. *NeurIPS*, 2019. 1

[3] Lama Alssum, Juan León Alcázar, Merey Ramazanova, Chen Zhao, and Bernard Ghanem. Just a glimpse: Rethinking temporal information for video continual learning. In *CVPRW*, 2023. 1, 2, 5, 6, 7

[4] Jihwan Bang, Heesu Kim, YoungJoon Yoo, Jung-Woo Ha, and Jonghyun Choi. Rainbow memory: Continual learning with a memory of diverse samples. In *CVPR*, 2021. 1, 3

[5] Jihwan Bang, Hyunseo Koh, Seulki Park, Hwanjun Song, Jung-Woo Ha, and Jonghyun Choi. Online continual learning on a contaminated data stream with blurry task boundaries. In *CVPR*, 2022. 3

[6] Federico Becattini, Tiberio Uricchio, Lorenzo Seidenari, Lamberto Ballan, and Alberto Del Bimbo. Am i done? predicting action progress in videos. *TOMM*, 16(4):1–24, 2020. 3

[7] Joao Carreira and Andrew Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *CVPR*, 2017. 1, 2, 5

[8] Matthias De Lange, Rahaf Aljundi, Marc Masana, Sarah Parisot, Xu Jia, Aleš Leonardis, Gregory Slabaugh, and Tinne Tuytelaars. A continual learning survey: Defying forgetting in classification tasks. *IEEE TPAMI*, 44(7):3366–3385, 2021. 1

[9] Guodong Ding, Fadime Sener, and Angela Yao. Temporal action segmentation: An analysis of modern techniques. *IEEE TPAMI*, 2023. 1, 2

[10] Guodong Ding and Angela Yao. Leveraging action affinity and continuity for semi-supervised temporal action segmentation. In *ECCV*, 2022. 2, 4

[11] Guodong Ding and Angela Yao. Temporal action segmentation with high-level complex activity labels. *IEEE TMM*, 2022. 2

[12] Yazan Abu Farha and Jurgen Gall. Ms-tcn: Multi-stage temporal convolutional network for action segmentation. In *CVPR*, 2019. 2, 3, 5, 6, 7

[13] Alireza Fathi, Xiaofeng Ren, and James M Rehg. Learning to recognize objects in egocentric activities. In *CVPR*, 2011. 5

[14] Mohsen Fayyaz and Jurgen Gall. Sct: Set constrained temporal transformer for set supervised action segmentation. In *CVPR*, 2020. 2

[15] Robert M French. Catastrophic forgetting in connectionist networks. *Trends in cognitive sciences*, 3(4):128–135, 1999. 1

[16] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *NeurIPS*, 2014. 3

[17] Tyler L Hayes, Kushal Kafle, Robik Shrestha, Manoj Acharya, and Christopher Kanan. Remind your neural network to prevent catastrophic forgetting. In *ECCV*, 2020. 1, 2

[18] Saihui Hou, Xinyu Pan, Chen Change Loy, Zilei Wang, and Dahua Lin. Learning a unified classifier incrementally via rebalancing. In *CVPR*, 2019. 1, 2

[19] Ahmet Iscen, Jeffrey Zhang, Svetlana Lazebnik, and Cordelia Schmid. Memory-efficient incremental learning through feature adaptation. In *ECCV*, 2020. 3

[20] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013. 3

[21] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017. 1, 2

[22] Hyunseo Koh, Dahyun Kim, Jung-Woo Ha, and Jonghyun Choi. Online continual learning on class incremental blurry task configuration with anytime inference. *arXiv preprint arXiv:2110.10031*, 2021. 3

[23] Yu Kong and Yun Fu. Human action recognition and prediction: A survey. *International Journal of Computer Vision*, 130(5):1366–1401, 2022. 1

[24] Hilde Kuehne, Ali Arslan, and Thomas Serre. The language of actions: Recovering the syntax and semantics of goal-directed human activities. In *CVPR*, 2014. 5

[25] Hilde Kuehne, Alexander Richard, and Juergen Gall. Weakly supervised learning of actions from transcripts. *Computer Vision and Image Understanding*, 163:78–89, 2017. 2

[26] Anna Kukleva, Hilde Kuehne, Fadime Sener, and Jurgen Gall. Unsupervised learning of action classes with continuous temporal embedding. In *CVPR*, 2019. 2, 5

[27] Colin Lea, Michael D Flynn, Rene Vidal, Austin Reiter, and Gregory D Hager. Temporal convolutional networks for action segmentation and detection. In *CVPR*, 2017. 2

[28] Timothée Lesort, Hugo Caselles-Dupré, Michael Garcia-Ortiz, Andrei Stoian, and David Filliat. Generative models from the perspective of continual learning. In *IJCNN*, 2019. 1, 2

[29] Jun Li and Sinisa Todorovic. Set-constrained viterbi for set-supervised action segmentation. In *CVPR*, 2020. 2

[30] Yingwei Li, Yi Li, and Nuno Vasconcelos. Resound: Towards action recognition without representation bias. In *ECCV*, 2018. 1

[31] Zhe Li, Yazan Abu Farha, and Jurgen Gall. Temporal action segmentation from timestamp supervision. In *CVPR*, 2021. 2

[32] Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE TPAMI*, 40(12):2935–2947, 2017. 1, 2

[33] Ji Lin, Chuang Gan, and Song Han. Tsm: Temporal shift module for efficient video understanding. In *ICCV*, 2019. 1

[34] David Lopez-Paz and Marc'Aurelio Ranzato. Gradient episodic memory for continual learning. *NeurIPS*, 2017. 1, 2

[35] Shervin Minaee, Yuri Boykov, Fatih Porikli, Antonio Plaza, Nasser Kehtarnavaz, and Demetri Terzopoulos. Image segmentation using deep learning: A survey. *IEEE TPAMI*, 44(7):3523–3542, 2021. 1

[36] Jaeyoo Park, Minsoo Kang, and Bohyung Han. Class-incremental learning for action recognition in videos. In *ICCV*, 2021. 1

[37] Yixuan Pei, Zhiwu Qing, Jun Cen, Xiang Wang, Shiwei Zhang, Yaxiong Wang, Mingqian Tang, Nong Sang, and Xueming Qian. Learning a condensed frame for memory-efficient video class-incremental learning. *NeurIPS*, 2022. 1

[38] Rahul Rahaman, Dipika Singhania, Alexandre Thiery, and Angela Yao. A generalized and robust framework for timestamp supervision in temporal action segmentation. In *ECCV*, 2022. 2

[39] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. icarl: Incremental classifier and representation learning. In *CVPR*, 2017. 1, 2, 3, 6

[40] Alexander Richard, Hilde Kuehne, and Juergen Gall. Action sets: Weakly supervised action segmentation without ordering constraints. In *CVPR*, 2018. 2

[41] Saquib Sarfraz, Naila Murray, Vivek Sharma, Ali Diba, Luc Van Gool, and Rainer Stiefelhagen. Temporally-weighted hierarchical clustering for unsupervised action segmentation. In *CVPR*, 2021. 2

[42] Fadime Sener and Angela Yao. Unsupervised learning and segmentation of complex activities from video. In *CVPR*, 2018. 2, 5

[43] Hanul Shin, Jung Kwon Lee, Jaehong Kim, and Jiwon Kim. Continual learning with deep generative replay. *NeurIPS*, 2017. 1, 2, 3

[44] Dipika Singhania, Rahul Rahaman, and Angela Yao. Iterative contrast-classify for semi-supervised temporal action segmentation. In *AAAI*, 2022. 2

[45] Dipika Singhania, Rahul Rahaman, and Angela Yao. C2f-tcn: A framework for semi-and fully-supervised temporal action segmentation. *IEEE TPAMI*, 2023. 2

[46] Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. Ucf101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv:1212.0402*, 2012. 1

[47] Yaser Souri, Yazan Abu Farha, Emad Bahrami, Gianpiero Francesca, and Juergen Gall. Robust action segmentation from timestamp supervision. In *BMVC*, 2022. 4

[48] Sebastian Stein and Stephen J McKenna. Combining embedded accelerometers with computer vision for recognizing food preparation activities. In *UbiComp*, 2013. 5

[49] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008. 8

[50] Andrés Villa, Kumail Alhamoud, Victor Escorcia, Fabian Caba, Juan León Alcázar, and Bernard Ghanem. vclimb: A novel video class incremental learning benchmark. In *CVPR*, 2022. 1, 2

[51] Limin Wang, Yuanjun Xiong, Zhe Wang, Yu Qiao, Dahua Lin, Xiaoou Tang, and Luc Van Gool. Temporal segment networks: Towards good practices for deep action recognition. In *ECCV*, 2016. 1

[52] Ye Xiang, Ying Fu, Pan Ji, and Hua Huang. Incremental learning using conditional adversarial networks. In *ICCV*, 2019. 1, 3

[53] Fangqiu Yi, Hongyu Wen, and Tingting Jiang. Asformer: Transformer for action segmentation. In *BMVC*, 2021. 2, 5, 6, 7