Learning to Predict Activity Progress by Self-Supervised Video Alignment

Gerard Donahue
Northeastern University
Boston, MA, USA
donahue.g@northeastern.edu

Ehsan Elhamifar
Northeastern University
Boston, MA, USA
e.elhamifar@northeastern.edu

Abstract

In this paper, we tackle the problem of self-supervised video alignment and activity progress prediction using in-the-wild videos. Our proposed self-supervised representation learning method carefully addresses different action orderings, redundant actions, and background frames to generate improved video representations compared to previous methods. Our model generalizes temporal cycle-consistency learning to allow for more flexibility in determining cycle-consistent neighbors. More specifically, to handle repeated actions, we propose a multi-neighbor cycle consistency and a multi-cycle-back regression loss by finding multiple soft nearest neighbors using a Gaussian Mixture Model. To handle background and redundant frames, we introduce a context-dependent drop function in our framework, discouraging the alignment of droppable frames. On the other hand, to learn from videos of multiple activities jointly, we propose a multi-head crosstask network, allowing us to embed a video and estimate progress without knowing its activity label. Experiments on multiple datasets show that our method outperforms the state-of-the-art for video alignment and progress prediction.[1]

1. Introduction

Understanding progress is central to a human's learning experience. We constantly modify and enhance our actions and behaviors based on our progress in different daily tasks [31]. Progress understanding is also fundamental for machines whose goal is to guide (e.g., assistive technologies) and seamlessly interact (e.g., robots) with humans [6, 35]. Imagine wearing augmented reality (AR) assistant glasses when assembling a complex machine, performing a medical procedure, or cooking a new hour-long recipe. Depending on the amount of progress made in the current step, the AR needs to show information about the current or next step(s). Additionally, progress is an important signal for computer vision tasks to run on AR task assistants, such as action segmentation and anticipation (depending on little/full progress in a step, subsequent frames will belong to the same/next step) [40, 52] and anomaly/error detection (a decrease in progress indicates an error) [34].

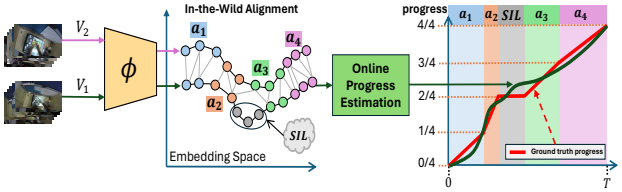


Figure 1. We develop a framework for self-supervised in-the-wild video alignment and online activity progress prediction.

Learning to predict progress, however, is challenging [8]. Progress can not be precisely calculated solely on the amount of time spent in an action (an expert often makes more progress than a novice within the same time window) and can be influenced by factors other than the user such as the environment (imagine cooking a recipe in a well-organized vs a cluttered kitchen). To estimate progress, we can break down an action into smaller phases and track where the user is within these phases (e.g., putting a coffee filter into a dripper involves taking a filter from its bag, folding it in half, folding it again into a quarter circle, creating a cone and inserting it into the dripper). Therefore, one approach for progress estimation is to label video clips of an action with its different phases and train a supervised phase classification model [3]. However, the ambiguity of defining a dictionary of fine-grained phases in advance and the high cost of annotating video frames with many action phases makes supervised learning impractical.

Self-supervised video alignment is an appealing alternative for progress prediction. It finds frame representations and associations between frames of different videos (ideally, aligning the same phases across videos), without requiring fine-grained human labeling. There has been a large body of recent literature addressing video alignment

[1]Code is publicly available at https://github.com/gerardDonahue/GTCC_CVPR2024.

using different approaches, such as temporal cycle consistency (TCC) [12], dynamic time warping [11, 16, 17] and optimal transport [39]. However, after aligning videos and learning the embedding space, progress prediction has been simply treated as fitting a linear regression model to the embeddings by assuming a linear progress model [12, 17, 39].

Limitations. Existing self-supervised methods have limitations for video alignment as well as progress prediction. First, when it comes to video alignment, the majority of existing works assume the same ordering of phases (monotonic), which is often violated in real videos (e.g., you can put a filter in a dripper and then put the dripper on the mug or vice versa). Additionally, some videos may have additional steps or background, which should not be aligned. Therefore, the recent approach in [39] (coined VAVA) addressed sequence alignment by considering background and redundant frames as well as non-monotonic ordering by imposing a temporal prior and an optimal prior on an optimal transport formulation, an intra-video contrastive term and an inter-video contrastive term. However, balancing multiple losses and handling repeated actions in the self-supervised setting is challenging and enforcing a temporal prior is unfit for in-the-wild alignment. Moreover, existing alignment techniques train distinct models for each action, resulting in substantial memory and computational overhead. This approach also complicates inference since activity/task identification is necessary, posing challenges particularly at the onset of activity/task execution.

Paper Contributions. In this paper, we develop a framework for self-supervised video alignment and online progress prediction that addresses the limitations of existing works, see Figure 1. We propose a generalization of TCC [12] to model background and redundant frames, as well as repetitive actions when performing alignment. More specifically, to handle repeated actions, we propose a multi-neighbor cycle consistency and a multi-cycle-back regression loss by finding multiple soft nearest neighbors using a Gaussian Mixture Model (GMM) instead of a single neighbor as in TCC. To handle background and redundant frames, we introduce a context-dependent drop function, discouraging the alignment of droppable frames.

On the other hand, to effectively align videos of multiple activities jointly and to better leverage their shared information while handling their distinctions, we propose a multi-head crosstask network, where each head automatically learns to focus on relevant components of video pairs. Thus, we can embed a video and estimate progress without knowing its activity label. Through extensive experiments on multiple datasets, we show the effectiveness of our framework for video alignment and progress estimation.

2. Related Works

Self-Supervised Video Representation Learning. Given the high annotation cost for video-based action learning and the need for copious amounts of data when training deep networks, many researchers have turned to self-supervised methods for embedding video frames [9, 28, 38, 43, 46, 63, 75]. Self-supervision has largely been popularized as a result of higher quality image embedding techniques trained on large-scale image datasets such as [10, 27, 37]. With better datasets, pre-trained image encoders, such as [18, 56–58], allow for more meaningful features that can enhance the performance of self-supervised representation learning methods. Many downstream tasks benefit from self-supervised video representation learning, such as action recognition [26, 74], temporal action segmentation [23, 33, 66, 68], step localization [14, 74], video retrieval [71], text-to-video retrieval [42], and audio-visual representation learning [25].

Video Alignment is the task of temporally synchronizing videos, such that similar frames or segments in separate videos can be mapped to each other. Traditional methods such as Dynamic Time Warping (DTW) [7, 11, 13, 16, 17, 48], or Canonical Correlation Analysis (CCA) [1, 21, 26, 49] have been used to align video frames. Many methods have adopted the use of Soft-DTW [7, 11, 16, 17, 41] or Soft-Restricted Edit Distance [51], where the gradient is well-defined and back-propogation is feasible. LAV [17] is a self-supervised video alignment method, which uses Soft-DTW in conjunction with temporal intra-video contrastive loss terms. Other contrastive approaches include [5, 39, 43, 50, 63, 69]. Drop-DTW [11] and DWSA [53] extend DTW by adding a "trash bucket" to the cost matrix for dynamic programming for background frame classification. Similarly, VAVA [39] added an additional bucket to the optimal transport formulation to align in-the-wild videos. VAVA also incorporated a bi-modal Gaussian prior on the optimal transport matrix, which allowed for the model to consider the temporality and optimality of the alignment. VAVA was shown to outperform previous methods for unconstrained procedural video datasets. Other methods capable of background detection but not video alignment include [4, 44, 54, 64, 65].

Cycle-consistency has been a pivotal technique in sequence alignment, involving the mapping of elements from one sequence to another, followed by an inverse mapping back to the original sequence. Beyond its application in video alignment, cycle-consistency finds utility in diverse domains, including 3D shape matching [22], depth estimation [15], co-segmentation [60, 61], and transivity in deep learning [15, 72, 73]. Cycle-consistency was first introduced for self-supervised video alignment in TCC [12] and showed good performance in aligning synchronizable

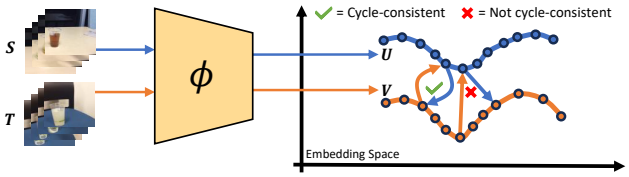


Figure 2. **Cycle-consistency Visualization.** Here, two videos are embedded by the alignment encoder, ϕ. With a green checkmark, a frame pair is shown to be cycle-consistent. With a red X-mark, a frame pair is shown to be not cycle-consistent.

videos, such as the Penn Action dataset [70] or the Pouring dataset [32]. Our method extends cycle-consistency to unconstrained videos, as observed in many datasets such as COIN [59] and EgoProceL [2].

Activity Progress Prediction is the task of predicting temporal progress in videos. While [3, 45] have studied a supervised approach for action progress prediction, they require temporal annotations of every action clip based on its phases, which is hard to define given the fine-grained nature of phases and cost to gather. Action completion [19, 20, 62, 71] is also related to progress, however it is only focused on classifying full progress achievement rather than predicting the progress at every time instant. In this work, we learn to predict progress at every frame in a self-supervised framework. Our proposed progress estimation technique also has applications in goal-based reinforcement learning, providing dense reward for learning [29, 67].

3. Problem Setting and Review

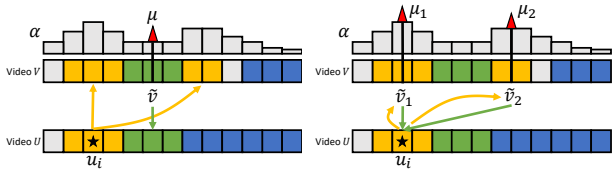
In this section, we discuss the problem of video alignment and review temporal cycle consistency (TCC) [12], which we build upon and generalize.

3.1. Video Alignment

Given two video sequences $\boldsymbol{S} = (\boldsymbol{s}_1, \dots, \boldsymbol{s}_N)$ and $\boldsymbol{T} = (\boldsymbol{t}_1, \dots, \boldsymbol{t}_M)$ with N and M frames, respectively, we denote their embeddings by $\boldsymbol{U} = (\boldsymbol{u}_1, \dots, \boldsymbol{u}_N)$ and $\boldsymbol{V} = (\boldsymbol{v}_1, \dots, \boldsymbol{v}_M)$. We use an encoder neural network ϕ to compute these embeddings, as depicted in Fig. 2. The goal of video alignment is to train the encoder, ϕ, such that if the two frames $\boldsymbol{s}_i$ and $\boldsymbol{t}_j$ represent the same action, the distance between their embeddings, $\boldsymbol{u}_i \in \mathbb{R}^D$ and $\boldsymbol{v}_j \in \mathbb{R}^D$, will be small, otherwise, the distance will be large.

3.2. Temporal Cycle Consistency (TCC)

TCC [12] finds a symmetric nearest neighbor embedding space by maximizing the number of *cycle-consistent* embeddings between two sequences. More specifically, consider a frame $\boldsymbol{u}_i \in \boldsymbol{U}$, and let its Euclidean nearest neighbor from $\boldsymbol{V}$ be denoted by $\boldsymbol{v}_j = \operatorname{argmin}_{\boldsymbol{v} \in \boldsymbol{V}} \|\boldsymbol{u}_i - \boldsymbol{v}\|$. Similarly, for a frame $\boldsymbol{v}_j \in \boldsymbol{V}$, denote its Euclidean nearest neighbor in $\boldsymbol{U}$ by $\boldsymbol{u}_k = \operatorname{argmin}_{\boldsymbol{u} \in \boldsymbol{U}} \|\boldsymbol{u} - \boldsymbol{v}_j\|$. The frame



(a) TCC (b) GTCC

Figure 3. **TCC Failure for Repeated Actions.** Common action segments of the sequences are shown using varying colors. $\boldsymbol{\alpha}$ is depicted based on the initial frame, $\boldsymbol{u}_i$, from the *yellow* action segment. TCC (3a) only allows a single $\tilde{\boldsymbol{v}}$ between both peaks in $\boldsymbol{\alpha}$. GTCC (3b) allows for two distinct neighbors, $\tilde{\boldsymbol{v}}_1$ and $\tilde{\boldsymbol{v}}_2$, which accurately represents repeated actions.

$\boldsymbol{u}_i$ is cycle-consistent if and only if $i = k$, see Fig. 2. TCC trains an encoder to embed frames such that the number of cycle-consistent frame embeddings is maximized across pairs of videos, hence, a symmetric nearest neighbor embedding space. Since argmin is non-differentiable, TCC adopts the softmax operation. Let Δ^M denote the probability simplex in M dimensions.[2] TCC begins by computing a soft nearest neighbor (SNN) of $\boldsymbol{u}_i$, denoted by $\tilde{\boldsymbol{v}}$,

$$\tilde{\boldsymbol{v}} = \sum_{j=1}^{M} \alpha_j \boldsymbol{v}_j, \quad \text{where,} \quad \alpha_j = \frac{e^{-\|\boldsymbol{u}_i - \boldsymbol{v}_j\|^2/\tau}}{\sum_{k=1}^{M} e^{-\|\boldsymbol{u}_i - \boldsymbol{v}_k\|^2/\tau}}. \quad (1)$$

Here $\boldsymbol{\alpha} = [\alpha_1, \dots, \alpha_M] \in \Delta^M$ is the *outgoing normalized similarity (forward distribution)* between $\boldsymbol{u}_i$ and embeddings in $\boldsymbol{V}$, and τ is the softmax temperature. TCC then uses the SNN, $\tilde{\boldsymbol{v}}$, to compute $\boldsymbol{\beta} = [\beta_1, \dots, \beta_N] \in \Delta^N$, which is the *incoming normalized similarity (backward distribution)* between $\tilde{\boldsymbol{v}}$ and frame embeddings in $\boldsymbol{U}$,

$$\beta_l = \frac{e^{-\|\boldsymbol{u}_l - \tilde{\boldsymbol{v}}\|^2/\tau}}{\sum_{j=1}^{N} e^{-\|\boldsymbol{u}_j - \tilde{\boldsymbol{v}}\|^2/\tau}}. \quad (2)$$

Cycle-consistency requires that $\boldsymbol{\beta}$ must have the highest probability mass at index i and very small mass at indices farther from i. Thus, TCC imposes a Gaussian distribution prior $\mathcal{N}(\mu_\beta, \sigma_\beta^2)$ on entries of $\boldsymbol{\beta}$ with mean $\mu_\beta = \sum_{l=1}^{N} \beta_l \cdot l$ and variance $\sigma_\beta^2 = \sum_{l=1}^{N} \beta_l \cdot (l - \mu_\beta)^2$. Requiring that the distribution mean be close to i with a small variance leads to minimizing a cycle-back regression loss, defined as

$$\mathcal{L}_{\text{cbr}} = \frac{(i - \mu_\beta)^2}{\sigma_\beta^2} + \lambda \log(\sigma_\beta), \quad (3)$$

where λ is the regularization weight. This leads to a differentiable loss that can be optimized using backpropagation.

4. Generalized Temporal Cycle Consistency (GTCC)

One of the advantages of TCC [12] compared to DTW-based methods [11, 17] is that it does not assume/enforce

[2] $\boldsymbol{\alpha} \in \Delta^M$ means that vector entries are nonnegative and sum to one.

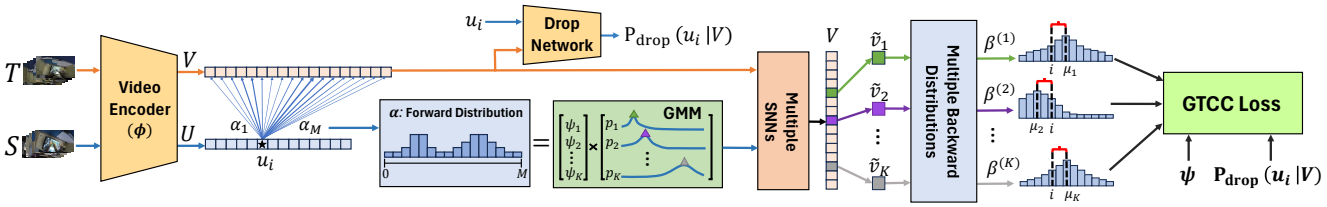


Figure 4. **GTCC.** We encode two videos into $\boldsymbol{U}$ and $\boldsymbol{V}$ and calculate the forward distribution, $\boldsymbol{\alpha}$. We use the output of the GMM optimization procedure and obtain K SNNs from K discrete Gaussians. We then compute $\boldsymbol{\beta}^{(k)}$ for each SNN. We also estimate the drop probability of each frame $\mathrm{P}_{\text{drop}}(\boldsymbol{u}_i|\boldsymbol{V})$ using a conditional network. Finally, we compute the final loss by weighting each cycle-back regression loss with weight from mixture densities, ψ, and modulate with the drop probabilities.

monotonic ordering of actions across videos. On the other hand, it has two major drawbacks when it comes to aligning in-the-wild videos.

First, TCC cannot handle background and additional actions, because it enforces cycle-consistency for every frame embedding. However, background actions (e.g., 'answering phone' when making coffee) or additional actions (e.g., the optional step of 'cleaning coffee filter') in one video but not in another should not be aligned.

Second, the TCC model assumes that each action only appears once in a video, therefore, cannot model action repetitions (e.g., repeating the actions of 'give breath' and 'give chest compression' when performing CPR). This is because computing the SNN in TCC is done with a unimodal Gaussian assumption on the similarity distributions, which is violated when an action is repeated in a video and leads to computing bad SNNs, see Fig. 3. We propose generalized temporal cycle consistency (GTCC), which addresses these limitations and aligns videos in the wild.

4.1. Multi-Cycle Consistency via Mixture SNNs

We propose multi-neighbor cycle consistency and a multi-cycle-back regression loss to handle repeated actions, see Fig. 4. For $\boldsymbol{u}_i \in \boldsymbol{U}$ belonging to an action a, a single SNN $\tilde{\boldsymbol{v}}$ obtained using (1) is restrictive when action a is repeated in the sequence $\boldsymbol{V}$. In such a case, as 3 shows, $\boldsymbol{u}_i$ will have multiple good nearest neighbors, yet the SNN $\tilde{\boldsymbol{v}}$ captures an average representation of them leading to a suboptimal solution. Therefore, we generalize TCC to allow for multiple SNNs and to assign an importance score to each SNN. This allows our model to find a set of SNNs based on a given $\boldsymbol{u}_i$, and to automatically gain awareness of their relative importance for promoting multiple-neighbor cycle-consistency.

For $\boldsymbol{u}_i \in \boldsymbol{U}$, we first obtain the outgoing distribution $\boldsymbol{\alpha} = [\alpha_1, \dots, \alpha_M]$, where α_j's are computed using (1). We then approximate $\boldsymbol{\alpha}$ with a discretized Gaussian Mixture Model (GMM) to capture different distribution modes corresponding to the same action. More specifically, we model

$$\alpha_j \approx p_j = \sum_{k=1}^{K} \psi_k p_j^{(k)}, \tag{4}$$

where $p_j^{(k)}$ is the outgoing probability of index j in video $\boldsymbol{V}$ for the k-th discretized Gaussian distribution $\mathcal{N}(\mu_{\alpha,k}, \sigma^2_{\alpha,k})$ and ψ_k denotes the weight of the mixture component k. Here, K is a hyperparameter defining the total number of mixture components. We typically set $K = 10$ in our experiments and as we show the results do not change much for different values of K as long as it is not very small. We find the parameters of the GMM, which are $\{\mu_k, \sigma_k^2, \psi_k\}_{k=1}^K$, by minimizing the KL divergence [55] between the two distributions $[\alpha_1, \dots, \alpha_M]$ and $[p_1, \dots, p_M]$. After finding the parameters of the GMM, we then compute one SNN for each mixture component,

$$\tilde{\boldsymbol{v}}^{(k)} = \sum_{j=1}^{M} p_j^{(k)} \cdot \boldsymbol{v}_j, \quad \forall k \in \{1, \dots, K\}. \tag{5}$$

With these K SNNs of $\boldsymbol{u}_i \in \boldsymbol{U}$, we then compute the incoming similarity distribution $\boldsymbol{\beta}^{(k)}$ for each SNN, $\tilde{\boldsymbol{v}}^{(k)}$. TCC assumes that throughout the sequence $\boldsymbol{U}$, $\boldsymbol{\beta}$ should only have a sharp peak at index i. For in-the-wild videos, this presents a challenge when $\boldsymbol{u}_i$ belongs to a repeated or non-contiguous action segment, where there would be more than one valid peak in the incoming distribution. To address this, we only enforce a local sharp peak around i for a discrete temporal window, w, where any index outside the i-centered window has zero probability (see supplementary materials for more details). For each $\boldsymbol{\beta}^{(k)}$, we compute the mean and variance over the indices of $\boldsymbol{U}$,

$$\mu_{\beta,k} = \sum_{l=1}^{N} \beta_l^{(k)} \cdot l, \quad \sigma^2_{\beta,k} = \sum_{l=1}^{N} \beta_l^{(k)} \cdot (l - \mu_{\beta,k})^2. \tag{6}$$

Finally, we compute one cycle-back regression loss for each mixture component as

$$\mathcal{L}_{\text{cbr}}^{(k)} = \frac{(i - \mu_{\beta,k})^2}{\sigma^2_{\beta,k}} + \lambda \cdot \log(\sigma_{\beta,k}). \tag{7}$$

Given that ψ_k is the relative importance of the k-th discrete Gaussian distribution, we build a multi-cycle back regression loss as

$$\mathcal{L}_{\text{multi-cbr}} = \sum_{k=1}^{K} \psi_k \cdot \mathcal{L}_{\text{cbr}}^{(k)}. \tag{8}$$

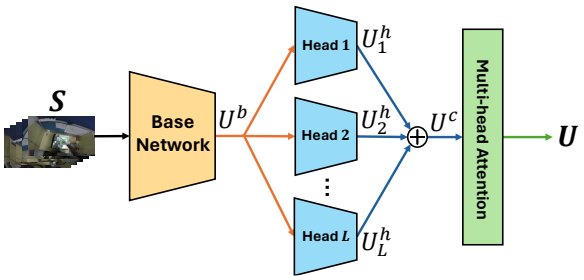


Figure 5. **Multi-head Crosstask Network (MCN).** Input frames, $\boldsymbol{S}$, are given to the base network to obtain base features, $\boldsymbol{U}^b$, which are subsequently given to L head networks that output intermediate video encodings, $\{\boldsymbol{U}_l^h\}_{l=1}^L$. We obtain the final video encoding $\boldsymbol{U}$ using attention over the outputs of the L heads.

4.2. Droppable Frame Detection

Another major limitation of TCC is that it cannot handle background and additional actions since it enforces cycle-consistency for all embeddings. GTCC addresses this issue by detecting frames in $\boldsymbol{U}$ to be dropped based on the frames in $\boldsymbol{V}$ and subsequently enforcing (i) multi-cycle consistency for alignable embeddings and (ii) bad alignment for droppable embeddings. Information (context) from video $\boldsymbol{V}$ is necessary when dropping frames in video $\boldsymbol{U}$ because the model can not know to drop frames corresponding to an additional action unless that action is not present in $\boldsymbol{V}$.

More specifically, we learn a drop-context function, $\boldsymbol{c} : \mathbb{R}^D \rightarrow \mathbb{R}^{D+1}$, characterized by a feedforward neural network, using all the embeddings $\boldsymbol{v}_j \in \boldsymbol{V}$ as

$$\boldsymbol{c}(\boldsymbol{V}) = \frac{1}{M}\sum_{j=1}^{M} \boldsymbol{c}(\boldsymbol{v}_j) \in \mathbb{R}^{D+1}. \tag{9}$$

GTCC uses the information in $\boldsymbol{V}$ to condition the drop detection in $\boldsymbol{U}$. More specifically, for each $\boldsymbol{u}_i \in \boldsymbol{U}$, we use a logistic regression model to compute the probability of dropping $\boldsymbol{u}_i$ given $\boldsymbol{V}$ as

$$\mathrm{P}_{\text{drop}}(\boldsymbol{u}_i|\boldsymbol{V}) = \sigma\big(\begin{bmatrix}\boldsymbol{u}_i^\top & 1\end{bmatrix}\boldsymbol{c}(\boldsymbol{V})\big), \tag{10}$$

where σ is the sigmoid function. We use the drop probability to build our final loss function, which we discuss next.

4.3. Learning GTCC

To learn the parameters of the embedding network ϕ and the drop-context model, we propose a loss so that alignable frames are encouraged to have *good* cycle-consistency and droppable frames are encouraged to have *bad* alignment. This dichotomy isolates the cycle-consistency property only to frames that are deemed 'alignable', and ensures frames deemed 'droppable' have no good nearest neighbors. We propose to minimize

$$\mathcal{L} = (1 - \mathrm{P}_{\text{drop}}(\boldsymbol{u}_i|\boldsymbol{V})) \cdot \mathcal{L}_{\text{multi-cbr}} + \frac{\mathrm{P}_{\text{drop}}(\boldsymbol{u}_i|\boldsymbol{V})}{\mathcal{L}_{\text{multi-cbr}}}, \tag{11}$$

where we minimize the multi-cycle-back regression loss for alignable frames for which $\mathrm{P}_{\text{drop}}(\boldsymbol{u}_i|\boldsymbol{V})$ is close to 0, and maximize the multi-cycle-back regression loss for droppable frames for which $\mathrm{P}_{\text{drop}}(\boldsymbol{u}_i|\boldsymbol{V})$ is close to 1. Notice one can also use $-\mathcal{L}_{\text{multi-cbr}}$ instead of $1/\mathcal{L}_{\text{multi-cbr}}$ in (11), however, the latter worked better in our experiments.

4.4. Online Progress Estimation

At inference time, given an online sequence up to frame $\boldsymbol{u}_i$, we want to estimate the progress, denoted by $\hat{\pi}(\boldsymbol{u}_i) \in [0, 1]$. We do so, by using the cumulative inter-frame geodesic distance on the frame features,

$$\hat{\pi}(\boldsymbol{u}_i) = \frac{\sum_{t=1}^{i-1} \|\boldsymbol{u}_{t+1} - \boldsymbol{u}_t\|}{\hat{G}}, \tag{12}$$

where $\hat{G}$ is ideally the total geodesic distance for the complete sequence (including past and future observations). Given that we do not have access to the future frames, we estimate $\hat{G}$ by averaging the total geodesic distances for all aligned training videos of the given task.

5. GTCC Embedding Network

Our goal is to train an embedding model that can leverage the shared information across related actions or tasks (e.g., making omelete and making scrambled eggs) for better feature learning and alignment of each task individually, while not requiring to know the action or task label of the video at inference time. Existing works, which train a separate model for each action/task, have neither of these capabilities. Therefore, we propose a new architecture, which we refer to as *Multi-head Crosstask Network* (MCN), shown in Fig. 5. MCN consists of a base network, multiple head networks, and an attention network. MCN base network is intended to output general useful features (across actions/tasks) to be used by multiple (L) head networks. Ideally, each head network specializes in aligning related phases of an action or steps of a task. The attention network takes all features from the head networks at each time instant and selects the relevant head and its features. See the supplemental material for more details.

6. Experiments

6.1. Experimental Setup

Datasets. We use four datasets for evaluations, two monotonic and two in-the-wild datasets. Similar to previous works [12, 17, 39], we report results on *Pouring* [32] and *Penn Action* [70] datasets, which are monotonic. *Pouring* contains videos of humans pouring liquid into a cup (e.g., orange juice poured into a blue cup), and *Penn Action* contains videos of humans completing athletic maneuvers (e.g., a woman serving a tennis ball). We also use *COIN* [59]

and *EgoProceL* [2], which are in-the-wild datasets, consisting of background frames, action repetitions, and different sequences of actions. EgoProceL is an egocentric dataset comprised of a collection of several datasets, including *CMU-MMAC* [30], *EGTEA Gaze+* [36], MECCANO [47], and EPIC-tent [24].

For Pouring, Penn Action, and COIN, we use 75% of videos for training and 25% for testing, while for EgoProceL we use 70% for training and 30% for testing since there are fewer videos per task. We also study two settings for evaluation. (i) Similar to prior works, we train a model separately for videos of each activity (single-task setting), whereas following other works we use an embedding network that corresponds to the base network in Fig. 5. (ii) We train a single model on videos of all activities jointly (multi-task setting). In Tab. 4, we test the multi-task setting using only the base network vs the full MCN architecture.

Evaluation Metrics. We use multiple metrics to evaluate the performance of different methods for alignment and progress prediction. For evaluating alignment, we use the existing metrics *Phase Classification* (PC) [12], *Enclosed Area Error* (EAE) [13] and *Kendall's Tau* (KT) [12]. PC measures how well the encoder can put frames belonging to the same action class together, thus a higher value is better. Similar to previous works, we use PC, where an SVM classifier is trained on 10%, 50%, and 100% of the ground-truth action labels. While PC measures how well different actions are separated from each other, it does not measure the quality of alignment within each action. Thus, we use Enclosed Area Error (EAE) that measures the alignment error within and across actions by comparing the nearest neighbors between two videos with their ground-truth alignment. When computing EAE for in-the-wild video pairs, we only include alignable frames in the computation. In cases where a frame has multiple valid neighbors, we calculate the smallest EAE among the set of neighbors. EAE is an error metric, therefore lower values indicate better alignment. See [13] for more details about EAE.

KT is a statistical measure that indicates how well-aligned two sequences are temporally. Given that KT is only suitable for monotonic sequences, we also propose a modified Kendall's Tau for in-the-wild datasets, which we call *Kendall's Tau in the Wild* (KTW). More specifically, we only evaluate KT on alignable frames and consider all variations of valid alignment for concordancy. This increases forgiveness when finding concordant pairs, because there may be many concordant neighbors for a frame in a redundant action segment. Therefore, KTW highlights the presence of disconcordant pairs.

To evaluate progress estimation, we use *PHase Progression* (PHP) [12] and introduce *Online Progress Error* (OPE). PHP trains a linear regressor over frame features to predict ground-truth progress values and computes the average R-squared measure over all videos. PHP assumes monotonicity in action sequences and, therefore, is only evaluated on the monotonic datasets. On the other hand, our online progress estimation in (12) applies to monotonic and in-the-wild datasets. Following [3], at each frame $\boldsymbol{u}_i$ we define a ground-truth progress value $\pi(\boldsymbol{u}_i) \in [0, 1]$ assuming that (i) each action segment contributes equally to the overall progress, and (ii) progress linearly increases throughout the duration of each action segment. We assume that silent frames do not contribute to progress, hence progress values remain constant during silent segments, see 6. We define OPE as the mean absolute difference between predicted progress and ground-truth online progress (lower is better).

Dataset	Model	PC ↑ 0.1	PC ↑ 0.5	PC ↑ 1.0	EAE ↓	KT ↑	PHP ↑	OPE ↓
Pouring	SAL	85.7	87.8	88.0	—	73.3	74.5	—
	TCN	89.2	90.4	90.4	—	86.7	80.6	—
	TCC	89.2	91.4	91.8	14.0	85.2	80.3	38.9
	LAV	91.6	**92.8**	92.8	9.0	85.6	80.5	31.4
	VAVA	**91.7**	91.8	92.5	29.6	87.6	83.6	37.8
	GTCC	71.2	89.2	**93.5**	**7.9**	**88.1**	**85.8**	**22.5**
Penn Action	SAL	74.9	78.3	80.0	—	63.4	59.4	—
	TCN	82.0	83.7	84.0	—	73.3	67.6	—
	TCC	81.3	83.4	84.5	18.2	73.5	67.3	35.5
	LAV	83.6	84.0	84.3	20.5	80.5	66.1	25.5
	VAVA	**83.9**	**84.0**	**84.5**	23.2	80.5	**70.9**	80.5
	GTCC	78.3	81.2	81.3	**16.7**	**88.3**	70.8	**14.5**

Table 1. Alignment and progress evaluations on **monotonic** datasets.

Dataset	Model	PC ↑ 0.1	PC ↑ 0.5	PC ↑ 1.0	EAE ↓	KTW ↑	OPE ↓
EgoProceL	TCC	66.0	67.4	68.3	21.5	13.9	34.0
	LAV	67.2	68.2	68.8	20.8	12.7	36.2
	VAVA	65.3	65.9	66.3	23.2	13.1	35.5
	GTCC	**75.0**	**78.9**	**80.6**	**13.3**	**14.0**	**13.6**
COIN	TCC	35.9	39.6	40.7	28.0	18.8	28.7
	LAV	36.8	38.9	39.8	26.7	23.2	22.7
	VAVA	43.8	46.2	47.3	32.0	22.2	27.6
	GTCC	**46.8**	**53.2**	**56.3**	**18.9**	**29.0**	**21.4**

Table 2. Alignment and progress evaluations on **in-the-wild** datasets.

Baselines. For the single-task setting, we compare our GTCC against SAL [43], TCN [50], TCC [12], LAV [17] and VAVA [39]. Among them, only VAVA is specifically designed for handling in-the-wild videos. For the multi-task setting, we compare GTCC with LAV and VAVA, which are state-of-the-art.

On the Penn Action, Pouring, and COIN datasets, for the single-task setting, we report the PC, PHP, and KT as previously reported in the most recent work [39]. Given that previous methods did not report EAE, KTW, and OPE, we re-implement all baselines on all datasets to report the performance based on these metrics. Similarly, we reproduce all numbers for the MCN multi-task (Table 4) and the EgoProceL experiments (Tables 2, 3, 4, 5, 6).

Implementation Details For the single-task setting, following all previous baselines, we use ResNet-50 [18] for the architecture of the encoder network. Please see [12] for more

Dataset	Model	Precision	Recall	F1	Accuracy
EgoProceL	VAVA	66.0	**98.4**	77.0	64.3
	GTCC	**67.5**	97.9	**78.2**	**68.9**
COIN	VAVA	41.7	78.8	53.5	44.7
	GTCC	**51.9**	**88.2**	**65.3**	**62.0**

Table 3. Evaluation of handling background actions (i.e., drop) of VAVA vs GTCC for the in-the-wild datasets.

Dataset	Method	Arch.	PC ↑	EAE ↓	KT ↑	KTW ↑	PHP ↑	OPE ↓
CMU-MMAC	LAV	ResNet	65.2	22.2	—	6.8	—	18.2
		MCN	69.8	**13.4**	—	12.1	—	16.1
	VAVA	ResNet	62.5	24.0	—	5.1	—	20.7
		MCN	63.7	19.7	—	11.4	—	17.9
	GTCC	ResNet	**70.4**	26.8	—	9.0	—	25.2
		MCN	67.8	14.9	—	**21.8**	—	**14.0**
EGTEA	LAV	ResNet	82.5	19.7	—	2.8	—	17.9
		MCN	82.1	11.4	—	38.7	—	22.4
	VAVA	ResNet	83.2	19.6	—	2.7	—	19.6
		MCN	82.3	10.7	—	38.1	—	19.7
	GTCC	ResNet	**83.6**	17.2	—	3.3	—	19.5
		MCN	82.5	**10.2**	—	**47.6**	—	**16.0**
Penn Action	LAV	ResNet	78.7	20.5	68.4	—	62.5	27.7
		MCN	85.3	5.7	94.3	—	69.0	**8.6**
	VAVA	ResNet	80.3	26.2	76.2	—	64.8	37.9
		MCN	85.4	5.9	94.4	—	72.6	14.3
	GTCC	ResNet	73.9	24.7	60.7	—	69.7	32.9
		MCN	**86.7**	**5.6**	**94.9**	—	**85.5**	9.4

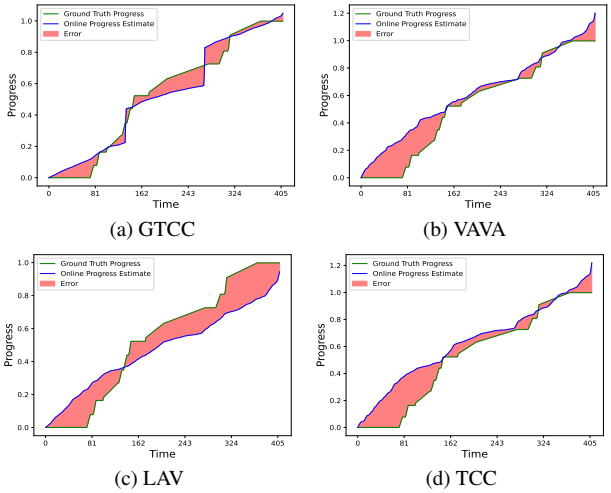
Table 4. Alignment and progress evaluation for the multi-task setting when training on ResNet-50 vs our proposed MCN architecture.

details regarding the encoder architecture. For the multi-task setting, we compare the ResNet-50 embedding network with our multi-head crosstask network (MCN), shown in Fig. 5, where the base network is ResNet-50 and all head and attention networks are 4-layer fully-connected feedforward network. For GTCC, we also simultaneously train a 4-layer feedforward neural network for the drop-context function, $\boldsymbol{c}$, in Sec. 4.2. We use $K = 15$ for in-the-wild datasets and $K = 5$ for monotonic datasets. For multi-task experiments using MCN, we set the number of attention heads to be the number of tasks in the dataset.

6.2. Experimental Results

Single-Task Setting Results. Table 1 and 2 show the alignment and progress estimation evaluation of different methods for monotonic and in-the-wild datasets, respectively. Notice that for the monotonic datasets in Table 1, GTCC achieves significantly better EAE (for alignment) and OPE (for progress) on both datasets, e.g., improving the OPE from 25.5% to 14.5% and EAE from 18.2% to 16.7% on Penn Action. Similarly, our method achieves higher KT values, especially on Penn Action, where it outperforms other methods by at least 7.8%. However, GTCC mostly does not improve the PC values of the baselines. This is because PC does not directly represent precise alignment but instead measures the effectiveness of clustering action segments.

On the other hand, as the results in Table 2 show,



(a) GTCC (b) VAVA

(c) LAV (d) TCC

Figure 6. Online progress estimation for GTCC (6a), VAVA (6b), LAV (6c) and TCC (6d) for a video from the task "making brownies" in the EgoProceL dataset.

GTCC significantly outperforms all baselines for all metrics. Specifically, GTCC improves the EAE by at least 7.5% on both EgoProceL and COIN and reduces OPE from 34.0% to 13.6% on EgoProceL and from 22.7% to 21.4% on COIN. Similarly, our method significantly improves the PC and KTW over the state-of-the-art. This is thanks to the fact that GTCC is specifically designed to handle in-the-wild videos that contain action repetitions, background activities, and action ordering variations.

Drop Performance. Table 3 compares the effectiveness of VAVA and GTCC for handling background activities on in-the-wild datasets for the single-task setting. We measure the precision, recall, F1 score, and accuracy of dropping background frames during pairwise alignment. Notice that on both datasets, GTCC has a higher F1 score and accuracy. Notably, GTCC achieves 62.0% accuracy on COIN compared to 44.7% by VAVA and obtains 68.9% accuracy on EgoProceL compared to 64.3% by VAVA. This shows the effectiveness of our context-based drop and alignment, which jointly learns alignable and droppable frames with a soft probabilistic approach.

Multi-Task Setting Results. In Table 4, we show the alignment and progress evaluations of LAV, VAVA, and GTCC for the multi-task setting, where we use all videos in each dataset to learn a single embedding network. We compare using ResNet-50 with using our proposed MCN architecture. Since GTCC outperformed TCC on progress and alignment metrics in the previous experiments, we do not report the results for TCC. As the results demonstrate, using ResNet for embedding in the multi-task setting leads to generally low performances for all methods. For example, on CMU-MMAC, KTW is less than 9% and EAE is more than 22% for all methods (similarly on EGTEA). On the other

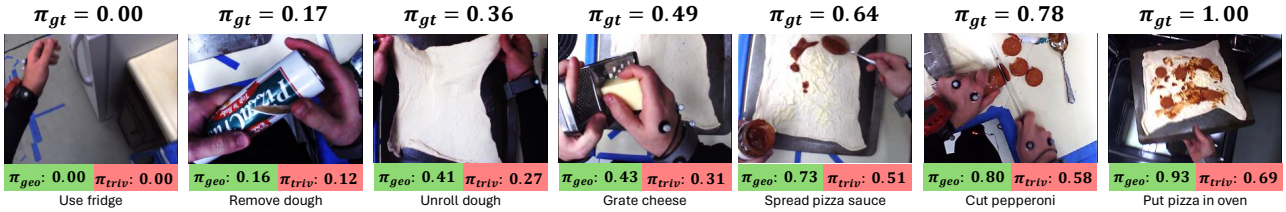


Figure 7. **Task Progression.** Frames from a video of the "pizza" task in EgoProceL. On top is the ground-truth progress (π_{gt}), in green is the geodesic progress from GTCC (π_{geo}), and in red is the trivial progress (π_{triv}) obtained by dividing test video time-steps by mean total video length from the train set.

	EAE ↓				OPE ↓				PC(1.0) ↑			
K	1	5	10	15	1	5	10	15	1	5	10	15
EgoP	19.4	19.2	**18.7**	19.4	20.4	**20.0**	20.3	20.9	73.9	74.8	**75.5**	74.3
COIN	23.2	21.4	22.2	**18.9**	25.5	23.1	22.0	**21.4**	**50.1**	49.6	48.9	46.8
Pour	11.6	**11.1**	11.7	11.5	**9.7**	12.3	10.9	10.2	**87.1**	84.9	84.9	86.3
Penn	26.7	**16.0**	16.1	16.1	**15.2**	16.7	17.8	16.7	**80.0**	78.2	78.22	78.9

Table 5. Effect of the number of GMM components, K, on GTCC.

hand, $\frac{6}{9}$ PC metrics, $\frac{9}{9}$ EAE metrics, $\frac{3}{3}$ KT metrics, $\frac{6}{6}$ KTW metrics, $\frac{3}{3}$ PHP metrics, and $\frac{7}{9}$ OPE metrics show improvement with the MCN architecture. Table 4 shows that when using MCN, the EAE, KT, KTW, and PHP metrics outperform 100% of the time, while PC and OPE outperform 67% and 78% of the time. This is because our architecture learns to specialize multiple heads for related actions using attention, while ResNet trains a single shared base model for all actions including the ones that are considerably different (e.g., serving tennis and jumping jack).

GMM Tuning and Ablations. Table 5 shows the effect of the number of mixture components, K, in the GMM on alignment and progress performance for GTCC. Notice that performance does not vary much under different values of K, showing the robustness of our approach. It is important to note that, as expected, $K = 1$ performs better comparatively on monotonic datasets than in-the-wild datasets.

Table 6 ablates the number of MCN heads (6a), and the drop term in GTCC (6b). We gather from 6a that alignment performance improves when the number of MCN heads scales alongside the number of tasks in the dataset (5 for both EGTEA Gaze+ and CMU-MMAC, which have 5 tasks). Finally, we see in 6b that the removal of the drop term leads to degraded performance, further showing the importance of the drop term in GTCC.

Qualitative Results. In Fig. 6, we show the comparison of different methods for online progress estimation on a test video from the task "making brownies" from the EgoProceL dataset. For each method, we show the ground-truth progress vs progress estimated via (12), $\hat{\pi}$, using the learned embeddings. Notice that GTCC's estimation of the progress fits the ground-truth nicely, while other methods have a large gap. This is because GTCC can effectively drop frames that correspond to background, while obtaining an

Figure 8. **In-the-wild Frame Retrieval** from COIN dataset.

	PC(1.0) ↑			EAE ↓		
# Heads	1	5	20	1	5	20
CMU	66.7	**67.8**	64.7	29.5	**14.9**	29.0
EGTEA	82.3	**82.5**	82.0	15.5	**10.2**	19.3

(a) Number of MCN heads.

	PC(1.0) ↑		EAE ↓	
Drop	✓	×	✓	×
Ego	**80.6**	65.2	**13.3**	24.3
COIN	**56.3**	46.6	**18.9**	27.9

(b) With (✓), without (×) dropout

Table 6. Ablation studies on (a) Number of MCN heads, (b) Drop.

embedding space where different actions are well-aligned. In Fig. 7, we show the ground-truth, trivial estimate via frame-counting, and $\hat{\pi}$ by GTCC from a EgoProceL video. Notice GTCC can effectively predict the progress online and better than the trivial estimate from frame-counting. In Fig. 8, we show results for in-the-wild frame retrieval for GTCC on the COIN dataset. This figure showcases the ability of GTCC to align frames with high visual similarity.

7. Conclusions

We studied a self-supervised approach, based on a proposed generalized temporal cycle consistency (GTCC) method, for video alignment and progress prediction, coupled with a novel attention-based architecture for multi-task learning. Our extensive experiments show that GTCC outperforms the state of the art, especially for in-the-wild videos. Additionally, we highlight the efficacy of our MCN architecture for enhancing multi-task learning. The fusion of GTCC and MCN emerges as a competitive and versatile framework.

Acknowledgements

This work is sponsored by DARPA PTG (HR00112220001), NSF (IIS-2115110), ARO (W911NF2110276). Content does not necessarily reflect the position/policy of the Government.

References

[1] Galen Andrew, Raman Arora, Jeff Bilmes, and Karen Livescu. Deep canonical correlation analysis. In *Proceedings of the 30th International Conference on Machine Learning*, pages 1247–1255, Atlanta, Georgia, USA, 2013. PMLR. 2

[2] Siddhant Bansal, Chetan Arora, and C.V. Jawahar. My view is the best view: Procedure learning from egocentric videos. In *European Conference on Computer Vision (ECCV)*, 2022. 3, 6

[3] Federico Becattini, Tiberio Uricchio, Lamberto Ballan, Lorenzo Seidenari, and A. Bimbo. Am i done? predicting action progress in videos. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, 16:1 – 24, 2017. 1, 3, 6

[4] Congqi Cao, Yajuan Li, Qinyi Lv, Peng Wang, and Yanning Zhang. Few-shot action recognition with implicit temporal alignment and pair similarity optimization. *Computer Vision and Image Understanding*, 210:103250, 2021. 2

[5] Minghao Chen, Fangyun Wei, Chong Li, and Deng Cai. Frame-wise action representations for long videos via sequence contrastive learning. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 13791–13800, 2022. 2

[6] Frederick G. Conrad, Mick P. Couper, Roger Tourangeau, and Andy Peytchev. The impact of progress indicators on task completion. *Interacting with Computers*, 22(5):417–427, 2010. 1

[7] M. Cuturi and M. Blondel. Soft-dtw: a differentiable loss function for time-series. *International Conference on Machine learning*, 2017. 2

[8] Frans de Boer, Jan C. van Gemert, Jouke Dijkstra, and Silvia L. Pintea. Is there progress in activity progress prediction?, 2023. 1

[9] Andong Deng, Taojiannan Yang, and Chen Chen. A large-scale study of spatiotemporal representation learning with a new benchmark on action recognition. *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 20462–20474, 2023. 2

[10] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009. 2

[11] Nikita Dvornik, Isma Hadji, Konstantinos G Derpanis, Animesh Garg, and Allan D Jepson. Drop-dtw: Aligning common signal between sequences while dropping outliers. *Neural Information Processing Systems*, 2021. 2, 3

[12] D. Dwibedi, Y. Aytar, J. Tompson, P. Sermanet, and A. Zisserman. Temporal cycle-consistency learning. *IEEE Conference on Computer Vision and Pattern Recognition*, 2019. 2, 3, 5, 6

[13] Niloufar Fakhfour, Mohammad ShahverdiKondori, and Hoda Mohammadzade. Video alignment using unsupervised learning of local and global features, 2023. 2, 6

[14] Junyu Gao, Mengyuan Chen, and Changsheng Xu. Fine-grained temporal contrastive learning for weakly-supervised temporal action localization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19999–20009, 2022. 2

[15] Clément Godard, Oisin Mac Aodha, and Gabriel J. Brostow. Unsupervised monocular depth estimation with left-right consistency. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6602–6611, 2017. 2

[16] Isma Hadji, Konstantinos G. Derpanis, and Allan D. Jepson. Representation learning via global temporal alignment and cycle-consistency. *IEEE Conference on Computer Vision and Pattern Recognition*, 2021. 2

[17] Sanjay Haresh, Sateesh Kumar, Huseyin Coskun, Shahram N. Syed, Andrey Konin, M. Zeeshan Zia, and Quoc-Huy Tran. Learning by aligning videos in time. *IEEE Conference on Computer Vision and Pattern Recognition*, 2021. 2, 3, 5, 6

[18] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016. 2, 6

[19] Farnoosh Heidarivincheh, Majid Mirmehdi, and Dima Damen. Action completion: A temporal model for moment detection, 2018. 3

[20] Farnoosh Heidarivincheh, Majid Mirmehdi, and Dima Damen. Weakly-supervised completion moment detection using temporal attention. In *2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*, pages 1188–1196, 2019. 3

[21] Harold Hotelling. Relations between two sets of variates. *Biometrika*, 28(3/4):321–377, 1936. 2

[22] Qixing Huang and Leonidas Guibas. Consistent shape maps via semidefinite programming. *Computer Graphics Forum*, 32, 2013. 2

[23] Yifei Huang, Yusuke Sugano, and Yoichi Sato. Improving action segmentation via graph-based temporal reasoning. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 14021–14031, 2020. 2

[24] Youngkyoon Jang, Brian Sullivan, Casimir Ludwig, Iain Gilchrist, Dima Damen, and Walterio Mayol-Cuevas. Epic-tent: An egocentric video dataset for camping tent assembly. In *International Conference on Computer Vision (ICCV) Workshops*, 2019. 6

[25] Simon Jenni, Alexander Black, and John Collomosse. Audio-visual contrastive learning with temporal self-supervision. In *Proceedings of the Thirty-Seventh AAAI Conference on Artificial Intelligence and Thirty-Fifth Conference on Innovative Applications of Artificial Intelligence and Thirteenth Symposium on Educational Advances in Artificial Intelligence*. AAAI Press, 2023. 2

[26] Tae-Kyun Kim, Shu-Fai Wong, and Roberto Cipolla. Tensor canonical correlation analysis for action classification. In *2007 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, 2007. 2

[27] Alex Krizhevsky. Learning multiple layers of features from tiny images. 2009. 2

[28] Akash Kumar, Ashlesha Kumar, Vibhav Vineet, and Yogesh Singh Rawat. Benchmarking self-supervised video representation learning, 2023. 2

[29] Sateesh Kumar, Jonathan Zamora, Nicklas Hansen, Rishabh Jangir, and Xiaolong Wang. Graph inverse reinforcement learning from diverse videos. In *Proceedings of The 6th Conference on Robot Learning*, pages 55–66. PMLR, 2023. 3

[30] Fernando De la Torre, Jessica K. Hodgins, Adam W. Bargteil, Xavier Martin, J. Robert Macey, Alex Tusell Collado, and Pep Beltran. Guide to the carnegie mellon university multimodal activity (cmu-mmac) database. Technical report, 2008. 6

[31] John W Lawrence, Charles S Carver, and Michael F Scheier. Velocity toward goal attainment in immediate experience as a determinant of affect. *Journal of Applied Social Psychology*, 32(4):788–802, 2002. 1

[32] Kimin Lee, Jaehyung Kim, Song Chong, and Jinwoo Shin. Making stochastic neural networks from deterministic ones, 2017. 3, 5

[33] Pilhyeon Lee, Jinglu Wang, Yan Lu, and Hyeran Byun. Weakly-supervised temporal action localization by uncertainty modeling. In *AAAI*, 2021. 2

[34] S. Lee, Z. Lu, Z. Zhang, M. Hoai, and E. Elhamifar. Error detection in egocentric procedural task videos. *IEEE Conference on Computer Vision and Pattern Recognition*, 2024. 1

[35] Shasha Li and Zhongzhen Lin. The impact of progress indicators and information density on users' temporal perception and user experience in mobile pedestrian navigation applications. *Displays*, 82:102603, 2024. 1

[36] Yin Li, Miao Liu, and James M. Rehg. In the eye of beholder: Joint learning of gaze and actions in first person video. In *Computer Vision – ECCV 2018: 15th European Conference, Munich, Germany, September 8–14, 2018, Proceedings, Part V*, page 639–655, Berlin, Heidelberg, 2018. Springer-Verlag. 6

[37] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft coco: Common objects in context. In *European Conference on Computer Vision*, 2014. 2

[38] Wei Lin, Xinghao Ding, Yue Huang, and Huanqiang Zeng. Self-supervised video-based action recognition with disturbances. *Trans. Img. Proc.*, 32:2493–2507, 2023. 2

[39] Weizhe Liu, Bugra Tekin, Huseyin Coskun, Vibhav Vineet, Pascal Fua, and Marc Pollefeys. Learning to align sequential actions in the wild. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2171–2181, 2022. 2, 5, 6

[40] Z. Lu and E. Elhamifar. Fact: Frame-action cross-attention temporal modeling for efficient action segmentation. *IEEE Conference on Computer Vision and Pattern Recognition*, 2024. 1

[41] Arthur Mensch and Mathieu Blondel. Differentiable dynamic programming for structured prediction and attention, 2018. 2

[42] Antoine Miech, Dimitri Zhukov, Jean-Baptiste Alayrac, Makarand Tapaswi, Ivan Laptev, and Josef Sivic. Howto100m: Learning a text-video embedding by watching hundred million narrated video clips, 2019. 2

[43] Ishan Misra, C. Lawrence Zitnick, and Martial Hebert. Shuffle and learn: Unsupervised learning using temporal order verification. In *European Conference on Computer Vision*, 2016. 2, 6

[44] N. Nakatsu, Y. Kambayashi, and S Yajima. A longest common subsequence algorithm suitable for similar text strings. pages 171–179, 1982. 2

[45] Davide Pucci, Federico Becattini, and Alberto Del Bimbo. Joint-based action progress prediction. *Sensors*, 23(1), 2023. 3

[46] Zhiwu Qing, Shiwei Zhang, Ziyuan Huang, Xiang Wang, Yuehuan Wang, Yiliang Lv, Changxin Gao, and Nong Sang. Mar: Masked autoencoders for efficient action recognition. *IEEE Transactions on Multimedia*, 26:218–233, 2024. 2

[47] Francesco Ragusa, Antonino Furnari, Salvatore Livatino, and Giovanni Maria Farinella. The meccano dataset: Understanding human-object interactions from egocentric videos in an industrial-like domain. In *Winter Conference on Applications of Computer Vision (WACV)*, 2021. 6

[48] David Sankoff, Joseph B. Kruskal, Joseph B Kru, and Mark Y. Liberman. Time warps, string edits, and macromolecules: the theory and practice of sequence comparison. 2016. 2

[49] Mehmet Sargin, Yuecel Yemez, Engin Erzin, and A. Tekalp. Audiovisual synchronization and fusion using canonical correlation analysis. *Multimedia, IEEE Transactions on*, 9:1396 – 1403, 2007. 2

[50] Pierre Sermanet, Corey Lynch, Jasmine Hsu, and Sergey Levine. Time-contrastive networks: Self-supervised learning from multi-view observation. In *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 486–487, 2017. 2, 6

[51] Y. Shen and E. Elhamifar. Semi-weakly-supervised learning of complex actions from instructional task videos. *IEEE Conference on Computer Vision and Pattern Recognition*, 2022. 2

[52] Y. Shen and E. Elhamifar. Progress-aware online action segmentation for egocentric procedural task videos. *IEEE Conference on Computer Vision and Pattern Recognition*, 2024. 1

[53] Y. Shen, L. Wang, and E. Elhamifar. Learning to segment actions from visual and language instructions via differentiable weak sequence alignment. *IEEE Conference on Computer Vision and Pattern Recognition*, 2021. 2

[54] Dingfeng Shi, Yujie Zhong, Qiong Cao, Lin Ma, Jia Li, and Dacheng Tao. Tridet: Temporal action detection with relative boundary modeling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 18857–18866, 2023. 2

[55] Jonathon Shlens. Notes on kullback-leibler divergence and likelihood. *CoRR*, abs/1404.2000, 2014. 4

[56] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014. 2

[57] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with

convolutions. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–9, 2015.

[58] Mingxing Tan and Quoc V. Le. Efficientnet: Rethinking model scaling for convolutional neural networks, 2020. 2

[59] Yansong Tang, Dajun Ding, Yongming Rao, Yu Zheng, Danyang Zhang, Lili Zhao, Jiwen Lu, and Jie Zhou. Coin: A large-scale dataset for comprehensive instructional video analysis. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 3, 5

[60] Fan Wang, Qixing Huang, and Leonidas J. Guibas. Image co-segmentation via consistent functional maps. In *Proceedings of the 2013 IEEE International Conference on Computer Vision*, page 849–856, USA, 2013. IEEE Computer Society. 2

[61] Fan Wang, Qixing Huang, Maks Ovsjanikov, and Leonidas J. Guibas. Unsupervised multi-class joint image segmentation. In *Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition*, page 3142–3149, USA, 2014. IEEE Computer Society. 2

[62] Yuanjun Xiong, Yue Zhao, Limin Wang, Dahua Lin, and Xiaoou Tang. A pursuit of temporal accuracy in general activity detection, 2017. 3

[63] Zihui Xue and Kristen Grauman. Learning fine-grained view-invariant representations from unpaired ego-exo videos via temporal alignment. *NeurIPS*, 2023. 2

[64] Xiang Yan, Syed Zulqarnain Gilani, Mingtao Feng, Liang Zhang, Hanlin Qin, and Ajmal Mian. Self-supervised learning to detect key frames in videos. *Sensors*, 20(23), 2020. 2

[65] Sakurai Yasushi, Faloutsos Christos, and Yamamuro Masashi. Stream monitoring under the time warping distance. pages 1046–1055, 2007. 2

[66] Serena Yeung, Olga Russakovsky, Ning Jin, Mykhaylo Andriluka, Greg Mori, and Li Fei-Fei. Every moment counts: Dense detailed labeling of actions in complex videos. *International Journal of Computer Vision*, 2017. 2

[67] Kevin Zakka, Andy Zeng, Pete Florence, Jonathan Tompson, Jeannette Bohg, and Debidatta Dwibedi. Xirl: Cross-embodiment inverse reinforcement learning. *Conference on Robot Learning (CoRL)*, 2021. 3

[68] Can Zhang, Meng Cao, Dongming Yang, Jie Chen, and Yuexian Zou. Cola: Weakly-supervised temporal action localization with snippet contrastive learning. In *CVPR*, 2021. 2

[69] Heng Zhang, Daqing Liu, Qi Zheng, and Bing Su. Modeling video as stochastic processes for fine-grained video representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2225–2234, 2023. 2

[70] Weiyu Zhang, Menglong Zhu, and Konstantinos G. Derpanis. From actemes to action: A strongly-supervised representation for detailed action understanding. In *2013 IEEE International Conference on Computer Vision*, pages 2248–2255, 2013. 3, 5

[71] Yue Zhao, Yuanjun Xiong, Limin Wang, Zhirong Wu, Xiaoou Tang, and Dahua Lin. Temporal action detection with structured segment networks. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 2933–2942, 2017. 2, 3

[72] Tinghui Zhou, Philipp Krähenbühl, Mathieu Aubry, Qixing Huang, and Alexei A. Efros. Learning dense correspondence via 3d-guided cycle consistency. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 117–126, 2016. 2

[73] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 2242–2251, 2017. 2

[74] Dimitri Zhukov, Jean-Baptiste Alayrac, Ramazan Gokberk Cinbis, David Fouhey, Ivan Laptev, and Josef Sivic. Cross-task weakly supervised learning from instructional videos. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3532–3540, 2019. 2

[75] Yuliang Zou, Jinwoo Choi, Qitong Wang, and Jia-Bin Huang. Learning representational invariances for data-efficient action recognition. *Computer Vision and Image Understanding*, 227:103597, 2023. 2