SVIP: Sequence Verification for Procedures in Videos

Yicheng Qian¹, Weixin Luo², Dongze Lian¹, Xu Tang³, Peilin Zhao⁴, Shenghua Gao¹
¹ShanghaiTech University, ²Meituan, ³Xiaohongshu Inc., ⁴Tencent AI Lab
{qianych, luowx, liandz, gaoshh}@shanghaitech.edu.cn, tangshen@xiaohongshu.com, masonzhao@tencent.com

Abstract

In this paper, we propose a novel sequence verification task that aims to distinguish positive video pairs performing the same action sequence from negative ones with step-level transformations but still conducting the same task. Such a challenging task resides in an open-set setting without prior action detection or segmentation that requires event-level or even frame-level annotations. To that end, we carefully reorganize two publicly available action-related datasets with step-procedure-task structure. To fully investigate the effectiveness of any method, we collect a scripted video dataset enumerating all kinds of step-level transformations in chemical experiments. Besides, a novel evaluation metric Weighted Distance Ratio is introduced to ensure equivalence for different step-level transformations during evaluation. In the end, a simple but effective baseline based on the transformer with a novel sequence alignment loss is introduced to better characterize long-term dependency between steps, which outperforms other action recognition methods. Codes and data will be released.

1. Introduction

In recent years, short-form videos filming people’s daily life widely disseminated on social media, which leads to the spurt of activity videos and greatly facilitated the research on video understanding [6, 14, 32, 56, 70, 86] as well. One can see from these videos that most daily activities are accomplished by serial steps instead of a single step. Such sequential steps form a procedure of which key-steps obey intrinsic consistency, while different participants may accomplish the same activity by different procedures with step-level divergence, as shown in Figure 7 (c). In this paper, we advocate a novel action task sequence verification intends to verify whether the procedures in two videos are step-level consistent, which can be applied to multiple potential tasks such as instructional training and performance scoring.

Why do we need sequence verification? Traditional action tasks such as action recognition [52, 91, 105], action localization [11, 50, 76] and action segmentation [24, 47, 101] have achieved significant progress due to the development of CNN as well as recently prevalent visual transformer [20]. However, most of these tasks follow a close-set setting with a limitation of predefined categories, illustrated in Figure 7 (a). Besides, accurate annotations of steps in numerous videos are extremely time-consuming and labor-intensive, followed by boundary ambiguities that have been studied in recent work [46, 73, 86, 108] though. However, our proposed sequence verification task circumvents both of these problems by verifying any video pair according to their distance in embedding space. In this way, the sequence
verification task neither requires the predefined labels nor consumes the intensive step annotations, which can easily handle the open-set setting.

As shown in Figure 7 (c), our proposed sequence verification aims to verify those procedures with semantic-similar steps rather than being associated with totally irrelevant tasks, which enables it to concentrate more on action step association rather than background distinction. Thus, an appropriate dataset is crucial to perform this task well. However, existing trimmed video datasets such as UCF101 [83], Kinetics [10], and Moments in Time [60] etc. are leveraged to carry out single-label action recognition. On the other hand, untrimmed video datasets like EPIC-KITCHENS [15], Breakfast [43], Hollywood Extended [7], ActivityNet [8] provide videos composed by multiple sub-actions and the corresponding step annotations, but they do not collect videos that especially performs similar or identical procedures. Thus, they cannot be used directly for sequence verification. To this end, we rearrange some datasets such as COIN [86] and Diving48 [51] where each task contains multiple videos recording different procedures, and each video has step-level annotations. Generally, videos with the same procedure are assigned to an individual category for training. Positive pairs and negative ones for testing are collected within the same procedure and cross different procedures in the same task, respectively. It should be noticed that these unscripted videos in the same procedure could be with a large appearance variance due to background divergence and personal preference, which makes sequence verification more challenging. Apart from that, we introduce a scripted filming dataset performing chemical procedures, where it includes all kinds of step-level transformations such as deletions, additions, and order exchanges. Thus, the effectiveness of any algorithm can be well justified by this newly proposed dataset. Additionally, since more step transformations may lead to larger feature distances which is unfair compared to less ones, we introduce a new evaluation metric Weighted Distance Ratio to make sure that every negative pair will be counted equally regardless of its step-level difference during evaluation.

As an unprecedented task, sequence verification may be solved by off-the-shelf action detectors [10, 13, 25, 30, 39, 49, 58, 63–65, 78, 84, 85, 95, 97]. However, their performance on Charades [79] or AVA [32], shown in Figure 7 (b), is not satisfactory to conduct step-level detection before verification. Although [3, 5, 11, 28, 53, 54, 75, 76, 92, 96–98, 100, 102, 103] perform well on ActivityNet [8] or THUMOS14 [38], it lacks persuasion since the two datasets either contain a few action classes or action instances per video. Thus, we introduce a simple but effective baseline CosAlignment Transformer (abbreviated as CAT), which leverages 2D convolution to extract discriminative features from sampled frames and utilizes a transformer to model inter-step temporal cor-

erelation in a video clip. Whereas representing the whole video with multiple steps as a single feature vector may lose information corresponding to the order of steps in a procedure. Thus, we introduce a sequence alignment loss that aligns each step in a positive video pair. The results show that our proposed method significantly outperforms other action recognition methods in our proposed sequence verification task.

We summarize our contributions as follows:

i) Problem setting: We propose a new task, sequence verification. To our knowledge, this is the first task focusing on procedure-level verification between videos.

ii) Benchmark: We rearrange two unscripted video datasets with significant diversity and propose a new scripted dataset with multiple step-level transformations to support this task. Moreover, a new evaluation metric is introduced especially for this novel task.

iii) Technical contributions: We propose a simple but effective baseline that contains a transformer to explicitly model the correlations between steps. Besides, a sequence alignment loss is introduced to improve the sensitivity to step disorder and absence. This novel baseline significantly outperforms other action recognition methods.

2. Related Work

Action Tasks. Traditional action-related tasks such as action recognition, action detection, and action segmentation have been greatly developed due to the advances in CNNs. i) As a means of general video representation, deep-learning-based action recognition can be generally summarized to stream-based methods [10, 12, 17, 19, 26, 52, 66, 80, 88, 89, 91, 105] and skeleton-based [21, 81, 93, 99] methods. Both kinds of methods aim to produce a feature representation for each trimmed video, to which a video-level label over predefined action categories is predicted according. ii) To seek the interested sub-actions in untrimmed videos, action detection [13, 25, 39, 53, 54, 58, 63–65, 84, 97, 104] are proposed to detect the start and end of sub-action instances and predict their categories. iii) To conduct dense action predictions in untrimmed videos, action segmentation is designed to label each frame including background in videos. With dense annotations, fully-supervised methods [23, 48, 59, 67, 69, 74, 76, 100] rely on sliding windows, Markov models, or temporal convolutional networks to model the temporal relations. However, dense annotations in videos require expensive human-labors as well as consume much time, though the weakly-supervised methods [7, 18, 36, 45, 68] with order labels of actions only have achieved satisfactory performance. Last but not least, it still remains a concern if these action-related tasks are able to generalize well to unknown classes in the wild. Different from them, our task suffers no restriction from an open environment during inference, so it can easily tackle an open-set...
### Table 1. The statistical information of three datasets. It is listed with an order of training, testing, and validation.

| Dataset     | # Tasks | # Videos  | # Steps | # Procedures | # Split Videos | # Split Samples |
|-------------|---------|-----------|---------|--------------|----------------|-----------------|
| COIN-SV     | 36      | 2,114     | 749     | 37 / 268 / 285 | 1,221 / 451 / 442 | 21,741 / 1,000 / 400 |
| Diving48-SV | 1       | 16,997    | 24      | 20 / 20 / 8   | 6,035 / 7,938 / 3,024 | 50,000 / 1,000 / 400 |
| CSV         | 14      | 1,941     | 106     | 45 / 25 / -   | 902 / 1,039 / -   | 8,531 / 1,000 / -  |

### Video datasets. Multiple existing video datasets [1, 8, 15, 31, 40, 41, 44, 61, 79, 83] have been dominated on video understanding during a long period. To begin with, HMDB51 [44] and UCF101 [83] that contains 51 and 101 classes of actions, respectively, are introduced for action recognition. Next, Something-Something [31] collects 147 classes of interactions between humans and objects in daily life. In addition, ActivityNet [8] and Kinetics [41] collect videos from YouTube and builds large-scale action recognition datasets. Other datasets for instructional video summarization and analysis [69, 82, 86] that are annotated with texts and temporal boundaries of a series of steps, contributes to the understanding of language and vision. EPIC-KITCHENS dataset [15] collects the human actions such as washing glass or cutting bell pepper in kitchen scenes and targets at the first-person perspective to reflect people’s goals and motivations.

### Instructional videos analysis. Instructional videos are generally accompanied with explanations such as audio or narrations matching the timestamps of sequential actions, which has attracted the research interest in the video understanding community. For instance, step localization [59, 86, 106] as well as action segmentation [27, 29, 62, 86] in instructional videos have been widely studied in the early stage. With the growing attention paid to this research topic, various kinds of tasks related to instructional videos have been proposed, e.g., video captioning [37, 57, 87, 106] which generates the description of a video based on the actions and events, visual grounding [35, 77] which locates the target in an image according to the language description, and procedure learning [2, 22, 27, 72, 73, 106] which extracts key-steps in videos.

### 3. Data Preparation

Due to the intrinsic step-procedure-task structure in the publicly available datasets COIN [86], and Diving48 [51], we reorganize these two datasets to support our proposed sequence verification task focusing on verifying various step-level transformations. However, the procedures in the same task may lack enough diversity in these datasets to fully verify the effectiveness of our proposed method for sequence verification. Thus, we collect a novel scripted dataset Chemical Verification enumerating all kinds of procedures in the same task, which will be introduced in the following.

![Figure 2. Dataset illustration. COIN-SV contains 36 daily life tasks such as 'Replace rearview mirror glass', 'Make burger', leading to high background diversity. Diving48-SV and CSV consist of videos in the diving competition and chemical experiments scene, respectively. Every video in three datasets is categorized by the sequence of steps it performs, as known as the procedure in the video. Note that only COIN-SV provides temporal annotation for steps, while Diving48-SV and CSV only provide the procedure-level annotation for each video.](image)

The rest of this section will introduce the common structure of these three datasets after rearrangement and collection, the specific processing and basic information of each dataset, as well as the statistical information of them.

### 3.1. Common Structure

As shown in Figure 7 (c), each dataset used in this paper contains videos completing various tasks, e.g., the original COIN dataset contains 180 tasks common in daily life. In practice, each individual task can be accomplished by different procedures of which steps as atomic actions still obey certain orders. Meanwhile, the steps of two procedures with different task-orientations will not overlap each other at most times. Thus, we will not introduce sequence verification cross tasks for the sake of challenge in these datasets.

In the training phase, all the methods are trained by a multi-category classification, and videos performing the same procedure are classified into the same category. In the testing phase, we collect the videos from the same procedure to form positive pairs and the videos from different
procedures but still in the same task to form negative pairs.

3.2. COIN-SV

COIN [86] is a comprehensive instructional video dataset that contains 180 tasks such as 'Replace the door knob', 'Change the car tire', and 'Install a ceiling fan'. This recently proposed dataset is quite challenging due to its background diversity and even significant distinctions between videos of the same procedures, which benefits our proposed sequence verification task. In total, it contains 11827 videos over 4715 procedures, which means COIN is followed by a long-tail distribution where most procedures have one or two videos only. To facilitate the classifier training, we select those procedures with more than 20 videos for training. The rest procedures are assigned to the validation and testing sets randomly, followed by the above sampling strategy. Overall, we ensure the numbers of positive and negative pairs are even. Since the original split in this dataset is reorganized, we name it COIN-SV.

3.3. Diving48-SV

Diving48 [51] dataset records diving competition videos with 48 kinds of diving procedures standardized by the international federation FINA, which consists of around 18,000 trimmed videos. Each diving procedure is a sub-action sequence of one-step takeoff, two-step movements in flight, and one-step entry. In total, 16997 videos over 48 procedures are publicly available up to now. Obviously, this dataset is less challenging than COIN due to its dual background including a board, a pool, and spectators and less step-level divergence. We assign 20, 8, 20 procedures for the training, validation, and testing sets, respectively. Similar to COIN-SV, we name it Diving48-SV.

3.4. Chemical Sequence Verification

Since the videos in COIN-SV and Diving48-SV are gathered from the internet, it is difficult to list all kinds of step-level transformations without predefined scripts, which is crucial for the sequence verification task. To this end, we collect a new dataset named Chemical Sequence Verification (CSV) containing videos with all kinds of step-level transformations such as deletions, additions, and order exchanges. Concretely, volunteers from an egocentric perspective are asked to conduct chemical experiments with predefined scripts. In a word, the CSV dataset includes 14 tasks, and each consists of 5 procedures. We select 45 procedures for training and 25 procedures for testing. The data gathering, annotations, and statistics information can be found in the supplementary materials.

The statistics of these three datasets can be found in Table 1, and we also visualize some samples in Figure 9.

4. Method

Classical models for video action recognition [10,12,17, 19, 26, 66, 80, 88, 89, 91] aim to predict action categories without paying attention to action orders as many as possible due to simple frame feature aggregation such as pooling. Nevertheless, our task intends to verify two videos with large as well as subtle step-level transformations. For instance, a video performing A, then B, and finally C is treated as a negative sample to another video carrying out A and finally C, while both of them may be successfully predicted by a traditional action classifier. To fit our proposed task, we introduce two remedies over the traditional action classification during training: i) procedures rather than tasks in the training set are regarded as training classes, in order to enable the model to distinguish those procedures even with tiny step-level transformations in the same task; ii) since pooling over frame features may bring order insensitivity to the model, we remain the temporal dimension without any down-sampling operation and it is finally reshaped to the channel dimension, followed by a fully-connected layer with order-sensitivity.

4.1. Preliminary

For a certain dataset $D = \{(V_i, S_i)\}_{i=1}^n$, a set of $n$ video clips $V$ are given with corresponding procedure annotations $S$. Here we do not use the timestamp annotations of steps since action detectors will not be used in this paper. We denote the model as $f: \mathbb{R}^{3 \times H \times W \times K} \rightarrow \mathbb{R}^C$. $K$ is the number of sampled frames in a video. $H$ and $W$ are height and width, respectively. $C$ is the total number of procedures in the training set. Following the paradigm in face verification [4, 16, 71], we treat sequence verification as a classification task during training. In the testing phase, the embedding distance between two videos indicates the verification score of this pair.

4.2. Baseline

We utilize a ResNet [34] backbone followed by a fully-connected layer to aggregate temporal information and a softmax classification layer as our baseline. Following TSN [91], we divide each input video into $K$ segments ($K = 16$ in our experiments). One frame in each segment is randomly selected to form the input tensor $x \in \mathbb{R}^{3 \times H \times W \times K}$, which is fed into the backbone and outputs a tensor with a shape of $D \times K$ where $D$ is feature dimension. Then it is flattened into a vector for the sake of order-sensitivity. Finally, a procedure classifier with $C$ categories is appended for training.

4.3. Vision Transformer

Transformer [90] has achieved great success in Natural Language Process and it has been applied in multiple computer vision tasks such as image recognition [20, 94] and
object detection [9, 107]. To better characterize inter-step correlations, we integrate the vision transformer ViT [20] into the backbone by replacing the global average pooling. As Figure 8 describes, we firstly flatten the spatial feature maps and apply a trainable linear projection layer on these flattened vectors, resulting in feature vectors $E_i \in \mathbb{R}^{D}$, $i = 1, 2, \cdots, K$. Further, randomly initialized position embedding is added to retain order information. In conclusion, the input $I$ of a standard transformer encoder is defined as

$$I = [E_1; E_2; \cdots; E_K] + E_{pos}, \in \mathbb{R}^{K \times D}.$$  

(1)

### 4.4. Sequence Alignment (SA)

So far, our proposed method aims to extract a global video representation supervised by the procedure classifier. However, the step order in a procedure is especially important in sequence verification. To make sure two positive procedures are step-level consistent, we propose a Sequence Alignment (SA) loss that explicitly imposes feature consistency step-by-step. Specifically, we extract the last spatial feature maps in the backbone and use global average pooling to produce feature vectors for all the steps in a given positive pair $seq_1$ and $seq_2$. Then cosine similarity is calculated for all the step pairs, resulting in a correlation matrix:

$$corr_{ij} = \frac{f_{1i} \cdot f_{2j}}{\|f_{1i}\| \cdot \|f_{2j}\|},$$  

(2)

where $corr_{ij}$ denotes the similarity value at the $i$-th row and the $j$-th column of the matrix $corr$, while $f_{1i}$ and $f_{2j}$ represent the $i$-th feature of $seq_1$ and $j$-th feature of $seq_2$, respectively. Next, we perform a softmax function on each row of the similarity matrix to produce $corr^1$, whose the $i$-th row is composed of cosine similarities between the $i$-th feature of $seq_1$ and every feature of $seq_2$. Similarly, we perform a softmax function on each column of the similarity and produce $corr^2$. We average these two matrices and denote the result as $corr_{avg}$. As explained above, the diagonal values of $corr_{avg}$ are then expected to be close to 1 while other values are expected to be close to 0, since both $corr^1$ and $corr^2$ have been normalized by softmax. Mathematically, our proposed Sequence Alignment (SA) loss $L_{seq}$ can be defined as:

$$L_{seq} = \|1 - h(\frac{corr^1 + corr^2}{2})\|_1,$$  

(3)

where $1$ is a vector whose entries are all one and $h$ is a function to extract the diagonal entries of a matrix.

### 4.5. Training Loss

As aforementioned in Section 4.1, we train the network by a procedure classification loss $L_{cls}$ as follows.

$$L_{cls} = \sum_{i=1}^{n} \delta(f(V_i), S_i)$$  

(4)

Figure 3. Pipeline overview. a) Applying a 2D backbone on the sampled frames (colored in the input frames) to capture features of individual steps, which we called intra-step features. The output of this module is a sequence of feature maps that are temporally modeled from the sampled frames. b) Applying a vision transformer [20] to aggregate the sequential feature maps. Note that there is only an encoder but no decoder in this module since we only care about the high-level procedure features instead of the specific feature for each step after decoding. c) This module aims at imposing the two sequences of feature maps to be matched.
where $\delta$ is the cross-entropy function.

Now, we train the network by the procedure classification loss and the sequence alignment loss in an end-to-end manner. Thus, the total loss $L$ can be summarized in the following:

$$L = L_{cls} + \lambda L_{seq}$$  \hspace{1cm} (5)

Here $\lambda$ is a hyper-parameter that adjusts the proportion of the sequence alignment loss, which is set to 1 by default.

4.6. Testing phase

During inference, the goal of sequence verification is to distinguish positive pairs from negative pairs. We denote each pair as $P_i = (V_{i_1}, V_{i_2})$. The model takes each video in $P_i$ as input and produces one $d$-dimensional visual embedding before the classification layer, which is denoted by $f' : \mathbb{R}^{K \times H \times W \times 3} \rightarrow \mathbb{R}^{D'}$. Next, we calculate the normalized Euclidean distance between the two procedures in the embedding space and the verification score $y_i$ is defined according to this distance:

$$d_i = g(f'(V_{i_1}), f'(V_{i_2}))$$  \hspace{1cm} (6)

$$y_i = \begin{cases} 1, & d_i \leq \tau, \\ 0, & \text{otherwise}. \end{cases}$$  \hspace{1cm} (7)

Where $g$ is a function that does $l_2$ normalization over two embeddings firstly and then calculates their Euclidean distance, $\tau$ is a threshold to decide whether the procedures are consistent. $y = 1$ means the procedures in two videos are consistent, otherwise inconsistent.

5. Experiments

5.1. Experimental Details

Datasets and setup. We conduct experiments on COIN-SV, Diving48-SV, and CSV. The specific information of each dataset is available in Section 3. Since this novel task is proposed to solve the open-set setting, there exists no procedure-level overlapping among the training, validation, and testing sets. However, step-level overlapping is unavoidable since different procedures can still contain several common steps.

Implementation Details. The ResNet-50 we employ is pre-trained on Kinetics-400 [41] to avoid over-fitting, while the new layers adopt Kaiming uniform initialization [33]. The experiments are conducted on 4 NVIDIA TITAN RTX GPUs with batch size 16, a cosine learning rate scheduler with a base learning rate of 0.0001, and weight decay 0.01. Adam [42] is used to optimize the whole network. For efficiency, we resize the raw images to $180 \times 320$. We also leverage horizontal flip, cropping, and color jittering for data augmentation. The feature dimension $D'$ before the classifier layer is set to 128 for all experiments.

Baselines. Since we are the first to introduce the sequence verification task, there are no existing methods that are specially designed for this task. Considering that we learn video representation during training, which is similar to the action recognition task, we compare our proposed method with some advanced action recognition baselines. Specifically, we build the following baselines: (1) Random: it randomly assigns ‘consistent’ or ‘inconsistent’ with a uniform distribution. (2) TSN [91]: a classic and commonly used action recognition method. This is similar to our method, except it utilizes average pooling instead to aggregate the temporal information. (3) TRN [105]: it is an extended version of TSN, which replaces average pooling with an MLP block to perform the temporal fusion. (4) TSM [52]: It introduces a temporal shift module based on TSN, which enables the feature of the current frame to include the ones of its adjacent frames by shifting cross temporal dimension. (5) Video Swin Transformer [56]: It is recently proposed as an variant of Swin Transformer [55] in the domain of videos. Specifically, it segments a frame stream to several 3D tokens and aggregates features according to the window that shifted on tokens through spatial and temporal dimensions, which is similar in spirit to TSM.

Evaluation Metrics. (1) AUC. We adopt the Area Under ROC Curve (abbreviated as AUC) as one of the measurements, which is commonly used to evaluate the performance of face verification. Higher AUC denotes better performance. (2) WDR. It is short for Weighted Distance Ratio. To begin with, we calculate the mean embedding distance per unit Levenshtein distance for negative pairs in order to guarantee the equivalence of each pair during evaluation because larger step-level transformations always lead to larger embedding distance, discussed in Section 5.5. The mean embedding distance over positive pairs is then computed. In the end, we use the ratio between negative distance and positive distance, namely the Weighted Distance Ratio, as an indicator of performance for all methods. Obviously, its higher value means better performance the methods arrive at. Mathematically, we define WDR as:

$$WDR = \frac{\sum_{i=1}^{N} w_{d_i}/N}{\sum_{j=1}^{P} d_j/P},$$  \hspace{1cm} (8)

where $P$ and $N$ are number of positives and negatives, respectively. $d_i$ and $d_j$ can be easily calculated by Equation 6. $w_{d_i}$ is defined as:

$$w_{d_i} = \frac{d_i}{cd_i}$$  \hspace{1cm} (9)

where $cd_i$ represents the text Levenshtein distance of pair $i$. Levenshtein distance, defined as the minimum number of operations required to transform one string into the other can be used as a measurement of how different in terms of
steps two procedures are. More explanations and evaluations can be found in Section 5.5.

5.2. Comparison of Methods

The quantitative results of all the methods on the three datasets are shown in Table 2. We can find that our proposed CAT exceeds all other baselines in most cases, evaluated on the AUC metric. It is worth noticing that CAT does not achieve the best WDR in all datasets since the best model is selected by the highest AUC in the validation set. More discussions about the difference between AUC and WDR can be found in Section 5.5. Apart from that, AUC on COIN-SV is extremely inferior compared to the other two datasets, which indicates its significant challenge due to its complex background and procedure diversity. Surprisingly, Video Swin Transformer achieving the best performance on a broad range of video recognition benchmarks such as action recognition, is inferior to other baselines. We conjecture that it suffers from data insufficiency, which is one of the limitations of this paper. One can refer to Section 5.7 for more discussions.

5.3. Ablation Study

In this section, we investigate the effectiveness of the vision transformer and sequence alignment module. The experiments are conducted on the testing set of the CSV dataset if not specially stated. To perform ablation analyses, we gradually add the vision transformer and sequence alignment module to the ResNet-50. The results in Table 3 show that both vision transformer and sequence alignment module improve the AUC performance on three datasets. As for WDR, CAT achieves the best performance on COIN-SV and Diving48-SV but is inferior to the vanilla on CSV.

![Figure 4. Visualizations of embeddings predicted by different model structures. Three kinds of colored points represent different procedures with labels 1.1, 1.2, and 1.3, respectively. The values above each sub-figure are the averaged intra-procedure variance and the inter-procedure variance, respectively.](image)

Table 2. Comparison with action recognition methods on the testing set of COIN-SV, Diving48-SV, and CSV dataset.

| Method     | Pretrain | #Param(M) | COIN-SV Val | Test | Diving48-SV Val | Test | CSV Val | Test |
|------------|----------|-----------|-------------|------|----------------|------|---------|------|
| Random     | -        | -         | 50.00 / -   | 50.00 / - | 50.00 / -     | 50.00 / - | 50.00 / - | 50.00 / - |
| TSN [91]   | K-400    | 22.67     | 53.38 / 0.365 | 47.01 / 0.3999 | 91.00 / 1.0835 | 81.87 / 0.6707 | 59.85 / 0.3447 |
| TRN [105]  | K-400    | 23.74     | 54.92 / 0.3665 | **57.19** / 0.3719 | 90.17 / **1.1438** | 80.69 / 0.5876 | 80.32 / **0.4677** |
| TSM [52]   | K-400    | 22.67     | 52.12 / 0.2948 | 51.25 / 0.3872 | 89.41 / 1.0035 | 78.19 / 0.5531 | 62.38 / 0.3308 |
| Swin [56]  | K-400    | 26.66     | 47.27 / 0.3895 | 43.70 / 0.3495 | 89.35 / 1.1066 | 73.10 / 0.5316 | 54.06 / 0.3141 |
| CAT(ours)  | K-400    | 72.32     | **56.81** / **0.4005** | 51.13 / **0.4098** | **91.91** / **1.0642** | **83.11** / **0.6005** | **83.02** / **0.4193** |

| Dataset | +ViT | +SA | AUC (%) | WDR |
|---------|------|-----|---------|-----|
| COIN-SV | ✓    | ✓   | 52.31   | 0.3677 |
|         | ✓    | ✓   | 55.46   | 0.3839 |
|         | ✓    | ✓   | **56.81** | **0.4005** |
| Diving48-SV | ✓    | ✓   | 90.11   | 1.0093 |
|         | ✓    | ✓   | **91.91** | **1.0642** |
| CSV     | ✓    | ✓   | 81.97   | 0.4403 |
|         | ✓    | ✓   | **83.02** | **0.4193** |

Table 3. Comparison between different model structures of our method on the testing set of the CSV dataset.

![Figure 4. Visualizations of embeddings predicted by different model structures. Three kinds of colored points represent different procedures with labels 1.1, 1.2, and 1.3, respectively. The values above each sub-figure are the averaged intra-procedure variance and the inter-procedure variance, respectively.](image)

5.4. Performance on Different Splits

As mentioned above, the step-level transformation contains deletions, additions, and order exchange of steps. To verify the order-sensitivity of the sequence alignment module, we further re-divide the testing set of CSV into two splits of which one consists of video pairs containing step additions and deletions, and the other consists of pairs containing order exchange of steps, which refers to alter-number split and alter-order split, respectively.
Table 4. Results on different test splits. (metric: AUC (%))

| Test Split     | CAT w/o SA | CAT w/ SA       |
|----------------|------------|-----------------|
| alter-number   | 73.01      | 75.82 (+2.81)   |
| alter-order    | 80.24      | 86.32 (+6.08)   |

The results shown in Table 4 show that CAT without SA module achieves inferior performance on both alter-number split and alter-order split while introducing the SA module brings more performance gain on alter-order split, which strongly supports the motivation of our proposed SA module that enables the model to be more order-sensitive.

5.5. WDR Curve

To carefully explore the character of our proposed evaluation metric WDR, we conduct two experiments to study the relationship between embedding distance and Levenshtein distance, and the relationship between WDR and AUC. First of all, the curve in Figure 13 Left describes a fact that the embedding distance between two procedures increases when the step-level difference denoted by Levenshtein distance gets larger. It is not surprising because the step order is preserved to some extent by all the methods, and large modifications in procedures will lead to distinct embedding differences. Thus, negative pairs with large step-level transformations will dominate the evaluation, which is unfair to those with small ones. To remedy this issue, WDR is introduced and it aims to evaluate embedding distance with respect to unit step-level transformation. Additionally, the curve in Figure 13 Right proves the positive correlation between WDR and AUC. The proposed WDR is then expected to be a complementary measurement metric in this new task.

5.6. Scoring Demo

As a potential solution to action assessment, sequence verification is able to act as a judge to score two procedures with fine-grained divergence. Here we show a diving scoring demo in Figure 14. The procedure in $V_0$ is chosen as the standard reference. We then calculate the cosine similarity as the score between the standard and each candidate video.

One can easily tell that $V_1$ achieves the highest score since it performs the same procedure as $V_0$, while the scores of $V_2$ and $V_3$ decrease with the enlargement of their step-level difference compared to the standard.

5.7. Limitation and Impact

Though we have introduced two reorganized datasets and collected one scripted dataset for this new task, it still suffers from data insufficiency leading to an unsatisfactory performance on Video Swin Transformer. Also, it may prevent this promising task from being applied to the real application considering the generalization ability in the wild. Vision transformer is introduced to aggregate temporal information, but it brings a parameter explosion as shown in Table 2. We believe this work still needs to be well explored in terms of data scale and model simplification. Except for these limitations, this work may bring a debate about the definition of standard reference, but it can solved by including guidelines from authorities.

6. Conclusion

In this work, we advocate a novel and interesting task sequence verification developed to verify two procedures with step-level differences when performing the same task. Dataset especially designed for this task is expected to record variants of the reference procedure as many as possible by altering its step order, removing existing steps, and including extra steps. To that end, we reorganize two publicly available action-related datasets with step-procedure-task structure and collect an egocentric dataset asking volunteers to perform various scripted procedures. In addition to that, we develop a new evaluation metric that has been well verified to be a complement to the existing AUC metric. Our proposed transformer-based procedure classification method has been wildly studied and acts as an excellent baseline for this new task. Finally, we hope that this promising task could provide a novel insight for the video understanding community but also other related research.
Appendix

The appendix part contains two sections as the supplementary of the text, which is arranged as follows:

1). The first section contains some complementary information of our proposed CSV dataset, e.g., data gathering, annotations, and statistics information.

2). The second section gives more examples of scoring and a demo of another application, early warning.

A. CSV Dataset

The existing action datasets can hardly support our task due to the following reasons: i) some datasets focus on single actions and don’t provide procedure videos; ii) some other datasets which contain procedure videos target other tasks such as action segmentation and action localization, i.e., they focus on the understanding of a single video rather than the verification of two videos, which leads to the lack of videos for similar procedures. However, the verification task indubitably requires a great number of videos that perform similar but slightly different step sequences for training. For the above reasons, we collect a new action verification dataset to support our proposed task. In this section, we firstly describe the gathering process of the dataset, then give the annotation details of the videos, and finally demonstrate the statistical information of the dataset.

A.1. Data Gathering

The dataset is recorded with the participation of 82 volunteers, whose ages range from 21 to 28, for performing scripted action sequences. Considering the constraints of venues, props, and personnel, we record videos of participants first setting up the equipment to perform a chemical experiment and then conducting that experiment. The specific process of recording is divided into the following steps: i) we firstly predefine 14 chemical experiment tasks, each of which contains consists of 5 procedures with a few step-level divergences, which will be detailed stated in Sec. A.2; ii) the volunteers are required to remember these predefined operations and equip with a head-mounted camera (shown in Figure 7); iii) after the camera start working, the volunteers are asked to perform the predefined action sequences and put hands on the table or their sides when finished, and then the recording will be stopped. In this way, the integrity of procedures in the videos gets guaranteed.

Following the collected method of [15], we choose Go-Pro HERO4 Black with an adjustable mounting such that the camera device can adjust to an appropriate pose with the variance of wearers’ height, which provides multi-angle views and makes that each video contains interactions between the volunteers’ hands and apparatus on the same experiment table. Besides, to ensure the stability and quality of the video, the camera is connected to a monitoring tablet via Bluetooth in order to monitor the quality of the recorded video at any time. Once a mistake occurs, the video will be discarded and re-shot. When shooting, the camera is set to the linear field of view, 24fps, and the resolution of 1920×1080. Stereo audio is captured but discarded since almost all procedures proceed silently, and the sounds in videos will cause irrelevant noises.

A.2. Action Sequence Annotations

In order to cater to our objective of verifying similar procedures with few step-level differences, we design fourteen different tasks in chemical experiments, and each enumerates all step-level transformations, e.g., additions, deletions, order exchange of steps. A step is defined as an action-object interaction whose label is always a combination of a verb, a noun, and sometimes prepositions. We label all procedures as 1.1 ∼ 1.5, 2.1 ∼ 2.5, · · · , 14.1 ∼ 14.5, totally 14 tasks, 70 labels. Note that we annotate each video only with a single serial number indicating the category of the procedure in the video, but without any temporal annotations, including the start and end frame of steps. Take the first task of procedures which is about screwing the test tube onto the iron stand and pouring water into the test tube as an example.

- 1.1: take (up the iron clamp) - screw (the iron clamp) - take (up the test tube) - screw (the iron clamp) - take (up the iron clamp) - take (up the a test tube) - pour (the conical flask) - put (down the conical flask)
- 1.2: take (up the iron clamp) - take (the a test tube) - screw (the iron clamp) - screw (the iron clamp) - take (up the conical flask) - pour (the conical flask) - put (down the conical flask)
- 1.3: take (up the test tube) - take (up the conical flask) - pour (the conical flask) - put (down the conical flask)
Figure 8. Step statistics. We list all 106 steps with their frequency of occurrence.

- **take** (up the iron clamp) - **screw** (the iron clamp) - **screw** (the iron clamp)

- **1.4**: **take** (up the iron clamp) - **screw** (the iron clamp) - **take** (up the conical flask) - **put** (down the conical flask) - **take** (up the test tube) - **screw** (the iron clamp)

- **1.5**: **take** (up the iron clamp) - **screw** (the iron clamp) - **take** (up the test tube) - **screw** (the iron clamp) - **take** (up the conical flask) - **put** (down the conical flask)

As illustrated above, compared to the 1.1, 1.2 and 1.3 disturb the order of actions; 1.4 not only changes the order, but also deletes the pour action; and for 1.5, it inserts take - put actions into the standard one.

The first group of procedures, which is a microcosm of the whole dataset, shows that most procedures differ in step order. The reason that we are so concerned about the order is that most action sequences will be unmeaning, sometimes even dangerous, if the order changes. For example, it is meaningless or even ridiculous to apply soap to hands after finishing washing hands.

A.3. Statistical Information

Figure 9 shows some statistics of our dataset. As illustrated, we have 18 atomic-level actions with different frequencies in total, among which take and put are the two most common actions. This makes sense since taking up or putting down something is also extremely common in reality. By interacting one action with different objects, we have 106 steps in total (listed in Figure 8). The videos’ length varies from 5.63s to 58.43s due to the diversity in complex-
ity among procedures and individual differences of participants, such as movement habits, the memory of the action sequence as well as familiarity with the operations. Totally, we collect 960,458 images of over 1,941 videos across 70 different kinds of procedures. On average, each video lasts 20.58 seconds, contains 495.85 frames, and consists of 9.53 steps.

B. Demos

B.1. Scoring

In this section, we demonstrate more examples as the scoring demo, which is detailed in Section 5.6 of the main body of this paper. For each dataset, we show two positive and two negative pairs, a total of eight videos with their procedure label. We can find Figure 12 has different procedure annotation from Figure 13 and 14, since the original COIN dataset [86] has temporal annotation for each step but Div-ing48 [51] and CSV doesn’t. It is worth noticing that $V_3$ and $V_4$ in Figure 13 perform the same diving sequence but recorded from different directions of the athlete but still outputs a high matching score.

B.2. Early Warning

In addition to scoring, the sequence verification task can also be applied in early warning. The system is required to alarm whenever it detects the occurrence of an unexpected step. Thus, how to detect atomic-level actions in real-time and how to compare the incomplete input procedure with the complete reference procedure would be the main difficulties of this promising task, which is also our future research direction.

However, the main body of this paper is to solve the verification problem of two complete procedures, which we named off-line verification. Here, we simply extend it to on-line sequence verification, where we can verify whether the input procedure is consistent with the reference in an on-line video stream. We design the following baseline. We take videos with labels 1.1 and 1.4, which are performed by two participants $P_A$, $P_B$, for demonstration. According to the detailed illustration in Section A.2, sequence 1.4 and sequence 1.1 are the same in the first three steps but are different in the fourth step. Note that although the third step are the same, the objects they interact with are different. However, such differences may be difficult for the model to recognize due to the light transmittance of glass products. The following is the specific description of the on-line action verification baseline.

Given a $t$-frame test procedure $P_{\text{test}}$ and the corresponding reference procedure $P_0$, and assume that it takes similar time intervals for each individual to perform the same step (this assumption is the basis of the baseline). Then we can assume that $P_0[1 : t \pm k]$ (the first $t \pm k$ frames of the reference procedure) is expected to perform the same step-sequence as $P_{\text{test}}[1 : t]$ does if they are labeled the same, where $k$ is the time window size ($k = 30$ in our experiment). For each $P_0[1 : t + i], -k \leq i \leq k$, we calculate the $l_2$ distance between $P_{\text{test}}[1 : t]$ and $P_{\text{test}}[1 : t]$ in the feature space $f$ and average them over $2k + 1$ cases as followed:

$$\sum_{i=-k}^{k} \frac{\|f(P_{\text{test}}[1 : t]) - f(P_0[1 : t + i])\|^2}{2k + 1}$$

Specifically, we stipulate all the frames of procedure 1.1 performed by $P_A$ as the complete reference procedure, and the first 100/150/200/250/300 frames of procedure 1.4 performed by $P_B$ as incomplete test procedures, the temporal annotation of these frames are given in Figure 10.

Figure 11 shows our experimental results. The blue line represents for the calculated $l_2$ distance in the feature space $f$ between 1.4-$P_B$ and 1.1-$P_A$ with different number of in-
put frames. For the convenience of explanation, we notate the number of input test frames as $i$. When $i = 100$, the value of distance remains relatively low. This is because both the first 100 frames of $1.4-P_B$ and the similar amount of frames of $1.1-P_A$ perform the same steps. When $i = 150$, note that although the objects interacted by the third step (conical flask and test tube), such glass products are hard to distinguish by the model, which also leads to the small value of distance. When $i = 200$, the step in $1.4-P_B$ is significantly different from the step in $1.1-P_A$. Thus, the value of distance rises rapidly. Besides, the broken line goes higher when $i = 250$ or 300 since more unmatched steps are included. We can easily catch the unexpected step in an on-line video stream through the huge jump of the line.

According to above, when we choose an appropriate threshold of distance, the $1.4-P_B$ vs. $1.1-P_A$ pair is verified until the number of input frames achieves 200, the moment when the unmatched step occurs, which satisfies the requirement of on-line action verification. This section states a coarse mechanism for on-line action verification and evaluates a toy sample based on that, which can be applied in the field of early warning. We hope that this brick cast away can attract a jode, i.e., makes more researchers study this challenging but promising task.

References

[1] Sami Abu-El-Haija, Nisarg Kothari, Joonseok Lee, Paul Natsev, George Toderici, Balakrishnan Varadarajan, and Sudheendra Vijayanarasimhan. Youtube-8m: A large-scale video classification benchmark. arXiv preprint arXiv:1609.08675, 2016. 3
[2] Jean-Baptiste Alayrac, Piotr Bojanowski, Nishant Agrawal, Josef Sivic, Ivan Laptev, and Simon Lacoste-Julien. Unsupervised learning from narrated instruction videos. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 4575–4583, 2016. 3
[3] Humam Alwassel, Silvio Giancola, and Bernard Ghanem. Tsp: Temporally-sensitive pretraining of video encoders for localization tasks. In Proceedings of the IEEE/CVF International Conference on Computer Vision, pages 3173–3183, 2021. 2
[4] Brandon Amos, Bartosz Ludwiczuk, Mahadev Satyanarayanan, et al. Openface: A general-purpose face recognition library with mobile applications. CMU School of Computer Science, 6(2), 2016. 4
[5] Anurag Bagchi, Jazib Mahmood, Dolton Fernandes, and Ravi Kiran Sarvadevabhatla. Hear me out: Fusional approaches for audio augmented temporal action localization. arXiv preprint arXiv:2106.14118, 2021. 2
[6] Sara Beery, Guanhang Wu, Vivek Rathod, Ronny Votel, and Jonathan Huang. Context r-cnn: Long term temporal context for per-camera object detection. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 13075–13085, 2020. 1
[7] Piotr Bojanowski, Rémi Lajugie, Francis Bach, Ivan Laptev, Jean Ponce, Cordelia Schmid, and Josef Sivic. Weakly supervised action labeling in videos under ordering constraints. In European Conference on Computer Vision, pages 628–643. Springer, 2014. 2
[8] Fabian Caba Heilbron, Victor Escorcia, Bernard Ghanem, and Juan Carlos Niebles. Activitynet: A large-scale video benchmark for human activity understanding. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 961–970, 2015. 2, 3
[9] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. arXiv preprint arXiv:2005.12872, 2020. 5
[10] Joao Carreira and Andrew Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 6299–6308, 2017. 2, 4
[11] Yu-Wei Chao, Sudheendra Vijayanarasimhan, Bryan Seybold, David A Ross, Jia Deng, and Rahul Sukthankar. Re-thinking the faster r-cnn architecture for temporal action localization. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 1130–1139, 2018. 1, 2
[12] RPW Christoph and Feichtenhofer Axel Pinz. Spatiotemporal residual networks for video action recognition. Advances in Neural Information Processing Systems, pages 3468–3476, 2016. 2, 4
Diederik P Kingma and Jimmy Ba. Adam: A method for

Anna Kukleva, Hilde Kuehne, Fadime Sener, and Jurgen

Hilde Kuehne, Alexander Richard, and Juergen Gall.

Hildegard Kuehne, Hueihan Jhuang, Estíbaliz Garrote, Will Kay, Joao Carreira, Karen Simonyan, Brian Zhang, Xin Li, Tianwei Lin, Xiao Liu, Wangmeng Zuo, Chao Li, Haroon Idrees, Amir R Zamir, Yu-Gang Jiang, Alex Gor-

Colin Lea, Michael D Flynn, Rene Vidal, Austin Reiter, and Gregory D Hager. Temporal convolutional networks for action segmentation and detection. In proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 156–156, 2017. 1

Colin Lea, Austin Reiter, René Vidal, and Gregory D Hager. Segmental spatiotemporal cnns for fine-grained action segment-

Wei Li, Zehuan Yuan, Dashan Guo, Lei Huang, Xiangzhong Fang, and Changhu Wang. Deformable tube network for action detection in videos. arXiv preprint arXiv:1907.01847, 2019. 2

Xin Li, Tianwei Lin, Xiao Liu, Wangmeng Zuo, Chao Li, Xiang Long, Dongliang He, Fu Li, Shilei Wen, and Chuang Gan. Deep concept-wise temporal convolutional networks for action localization. In Proceedings of the 28th ACM International Conference on Multimedia, pages 4004–4012, 2020. 1

Yingwei Li, Yi Li, and Nuno Vasconcelos. Resound: Towards action recognition without representation bias. In Proceedings of the European Conference on Computer Vision (ECCV), pages 513–528, 2018. 2, 3, 4, 11

Ji Lin, Chuang Gan, and Song Han. Tsm: Temporal shift module for efficient video understanding. In Proceedings of the IEEE/CVF International Conference on Computer Vision, pages 7083–7093, 2019. 1, 2, 6, 7

Tianwei Lin, Xun Zhao, Haisheng Su, Chongjing Wang, and Ming Yang. Bsn: Boundary sensitive network for temporal action proposal generation. In Proceedings of the IEEE/CVF International Conference on Computer Vision, pages 3889–3898, 2019. 2

Tianwei Lin, Xin Liu, Xin Li, Errui Ding, and Shilei Wen. Bmn: Boundary-matching network for temporal action proposal generation. In Proceedings of the European Conference on Computer Vision (ECCV), pages 3–19, 2018. 2

Ze Liu, Yuting Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. arXiv preprint arXiv:2103.14050, 2021. 6

Ze Liu, Jia Ning, Yue Cao, Yixuan Wei, Zheng Zhang, Stephen Lin, and Han Hu. Video sin transformer. arXiv preprint arXiv:2106.13230, 2021. 1, 6, 7

Huaiashuo Luo, Lei Ji, Botian Shi, Haoyang Huang, Nan Duan, Tianrui Li, Jason Li, Taroon Bharti, and Ming Zhou. Univl: A unified video and language pre-training model for multimodal understanding and generation. arXiv preprint arXiv:2002.06353, 2020. 3

Effrosyni Mavroudi, Benjamín Béjar Haro, and René Vidal. Representation learning on visual-symbolic graphs for video understanding. In European Conference on Computer Vision, pages 71–90. Springer, 2020. 2

Antoine Miech, Jean-Baptiste Alayrac, Lucas Smaira, Ivan Laptev, Josef Sivic, and Andrew Zisserman. End-to-end learning of visual representations from uncurated instructional videos. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 9879–9889, 2020. 2, 3

Mathew Monfort, Alex Andonian, Bolei Zhou, Kandan Ramakrishnan, Sarah Adel Bargal, Tom Yan, Lisa Brown, Quanfu Fan, Dan Gutfriend, Carl Vondrick, et al. Moments in time dataset: one million videos for event understanding. IEEE transactions on pattern analysis and machine intelligence, 42(2):502–508, 2019. 2

Naila Murray, Luca Marchesotti, and Florent Perronnin. Ava: A large-scale database for aesthetic visual analysis. In 2012 IEEE Conference on Computer Vision and Pattern Recognition, pages 2408–2415. IEEE, 2012. 3

AJ Piergiovanni, Anelia Angelova, Michael S Ryoo, and Irfan Essa. Unsupervised action segmentation for instructional videos. arXiv preprint arXiv:2106.03738, 2021. 3
Du Tran, Lubomir Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. Learning spatiotemporal features with 3d convolutional networks. In Proceedings of the IEEE international conference on computer vision, pages 4489–4497, 2015. 2, 4

Du Tran, Jamie Ray, Zheng Shou, Shih-Fu Chang, and Manohar Paluri. Convnet architecture search for spatiotemporal feature learning. arXiv preprint arXiv:1708.05038, 2017. 2, 4

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In Advances in neural information processing systems, pages 5998–6008, 2017. 4

Limin Wang, Yuanjun Xiong, Zhe Wang, Yu Qiao, Dahua Lin, Xiaou Tang, and Luc Van Gool. Temporal segment networks: Towards good practices for deep action recognition. In European conference on computer vision, pages 20–36. Springer, 2016. 1, 2, 4, 6, 7

Xiang Wang, Zhiwu Qing, Ziyuan Huang, Yutong Feng, Shiwei Zhang, Jianwen Jiang, Mingqian Tang, Changxin Gao, and Nong Sang. Proposal relation network for temporal action detection. arXiv preprint arXiv:2106.11812, 2021. 2

Junwu Weng, Chaoqun Weng, and Junsong Yuan. Spatio-temporal naive-bayes nearest-neighbor (st-nbnn) for skeleton-based action recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 4171–4180, 2017. 2

Bichen Wu, Chenfeng Xu, Xiaoliang Dai, Alvin Wan, Peizhao Zhang, Masayoshi Tomizuka, Kurt Keutzer, and Peter Vajda. Visual transformers: Token-based image representation and processing for computer vision. arXiv preprint arXiv:2006.03677, 2020. 4

Chao-Yuan Wu, Christoph Feichtenhofer, Haoqi Fan, Mariohar Paluri. Convnet architecture search for spatiotemporal representation and processing for computer vision. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 284–293, 2019. 2

Yuanjun Xiong, Yue Zhao, Limin Wang, Dahua Lin, and Xiaou Tang. A pursuit of temporal accuracy in general activity detection. arXiv preprint arXiv:1703.02716, 2017. 2

Huijuan Xu, Abir Das, and Kate Saenko. R-c3d: Region convolutional 3d network for temporal activity detection. In Proceedings of the IEEE international conference on computer vision, pages 5783–5792, 2017. 2

Mengmeng Xu, Chen Zhao, David S Rojas, Ali Thabet, and Bernard Ghanem. G-tad: Sub-graph localization for temporal action detection. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 10156–10165, 2020. 2

Sijie Yan, Yuanjun Xiong, and Dahua Lin. Spatial temporal graph convolutional networks for skeleton-based action recognition. arXiv preprint arXiv:1801.07455, 2018. 2

[100] Serena Yeung, Olga Russakovsky, Greg Mori, and Li Fei-Fei. End-to-end learning of action detection from frame glimpses in videos. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 2678–2687, 2016. 2

[101] Fangqiu Yi, Hongyu Wen, and Tingting Jiang. Asformer: Transformer for action segmentation. In The British Machine Vision Conference (BMVC), 2021. 1

[102] Runhao Zeng, Wenbing Huang, Mingkui Tan, Yu Rong, Peilin Zhao, Junzhou Huang, and Chuang Gan. Graph convolutional networks for temporal action localization. In Proceedings of the IEEE/CVF International Conference on Computer Vision, pages 7094–7103, 2019. 2

[103] Chen Zhao, Ali K Thabet, and Bernard Ghanem. Video self-stitching graph network for temporal action localization. In Proceedings of the IEEE/CVF International Conference on Computer Vision, pages 13658–13667, 2021. 2

[104] Yue Zhao, Yuanjun Xiong, Limin Wang, Zhirong Wu, Xiaou Tang, and Dahua Lin. Temporal action detection with structured segment networks. In Proceedings of the IEEE International Conference on Computer Vision, pages 2914–2923, 2017. 2

[105] Bolei Zhou, Alex Andonian, Aude Oliva, and Antonio Torralba. Temporal relational reasoning in videos. In Proceedings of the European Conference on Computer Vision (ECCV), pages 803–818, 2018. 1, 2, 6, 7

[106] Luowei Zhou, Chenliang Xu, and Jason J Corso. Towards automatic learning of procedures from web instructional videos. In Thirty-Second AAAI Conference on Artificial Intelligence, 2018. 3

[107] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable detr: Deformable transformers for end-to-end object detection. arXiv preprint arXiv:2010.04159, 2020. 5

[108] Dimitri Zhukov, Jean-Baptiste Alayrac, Ramazan Gokberk Cinbis, David Fouhey, Ivan Laptev, and Josef Sivic. Cross-task weakly supervised learning from instructional videos. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 3537–3545, 2019. 1
Figure 12. COIN-SV scoring example.
Figure 13. Diving48-SV scoring example.
Figure 14. CSV scoring example.

$S(V_1, V_2) = 0.9253$

$S(V_3, V_4) = 0.8362$

$S(V_5, V_6) = 0.1051$

$S(V_7, V_8) = 0.1059$

$V_1$: take up the test tube - take up the conical flask - pour the conical flask - put down the conical flask - screw the iron clamp - take up the horn tube - insert the horn tube

$V_2$: take up the test tube - take up the conical flask - pour the conical flask - put down the conical flask - screw the iron clamp - take up the horn tube - insert the horn tube

$V_3$: take up the test tube - shake the test tube - put down the test tube - take up the dropper bottle - uncover the dropper bottle - squeeze the dropper - cover the dropper bottle with the dropper - put down the dropper bottle

$V_4$: take up the test tube - shake the test tube - put down the test tube - take up the dropper bottle - uncover the dropper bottle - squeeze the dropper - cover the dropper bottle with the dropper - put down the dropper bottle

$V_5$: put down the beaker - take up the reagent bottle - screw the reagent bottle cap - pour the reagent bottle - screw the reagent bottle cap - put down the reagent bottle

$V_6$: take up the reagent bottle - screw the reagent bottle cap - screw the reagent bottle cap - put down the reagent bottle - take up the glass rod - take up the beaker - stir the glass rod - put down the glass rod - put down the beaker

$V_7$: take up the jar - uncover the jar cap - pour the jar - cover the jar with the jar cap - put down the jar

$V_8$: take up the jar - uncover the jar cap - put down the jar - pour the jar - cover the jar with the jar cap