Unsupervised Discovery of Actions in Instructional Videos

AJ Piergiovanni[1]
ajpiergi@google.com
Anelia Angelova[1]
anelia@google.com
Michael S. Ryoo[2]
mryoo@google.com
Irfan Essa[1]

[1] Google Research
[2] Robotics at Google

Abstract

In this paper we address the problem of automatically discovering atomic actions from instructional videos. Instructional videos contain complex activities and are a rich source of information for intelligent agents, such as, autonomous robots or virtual assistants, which can, for example, automatically 'read' the steps from an instructional video and execute them. However, videos are rarely annotated with atomic activities, their boundaries or duration. We present an unsupervised approach to learn atomic actions of structured human tasks from a variety of instructional videos. We propose a sequential stochastic autoregressive model for temporal segmentation of videos, which learns to represent and discover the sequential relationship between different actions of the task, and provides automatic and unsupervised self-labeling. We evaluate on the breakfast, 50-salads and narrated instructional videos datasets. Code will be open sourced.

1 Introduction

Instructional videos cover a wide range of tasks: cooking, furniture assembly, repairs, etc. The availability of online instructional videos for almost any task provides a valuable resource for learning, especially in the case of learning robotic tasks. However, instructional videos are rarely annotated with atomic action-level instructions. Several works have studied weakly-supervised settings where the order or presence of actions per-video is given, but not their duration [12, 23]. In this work, we propose a method to learn to segment instructional videos in atomic actions in an unsupervised way, i.e., without any annotations. To do this, we take advantage of the structure in instructional videos: they comprise complex actions which inherently consist of smaller atomic actions with predictable order. While the temporal structure of activities in instructional videos is strong, there is high variability of the visual appearance of actions, which makes the task, especially in its unsupervised setting, very challenging. For example, videos of preparing a salad can be taken in very different environments, using kitchenware and ingredients of varying appearance.

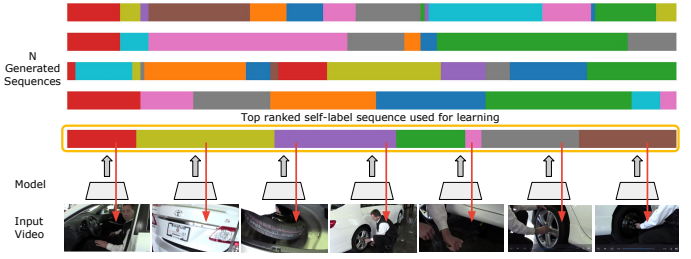

Figure 1: Overview: Our model generates multiple sequences for each video which are ranked based on several constraints (colors represent different actions). The top ranked sequence is used as self-labels to train the action segmentation model. This processes is repeated until convergence. No annotations are used.

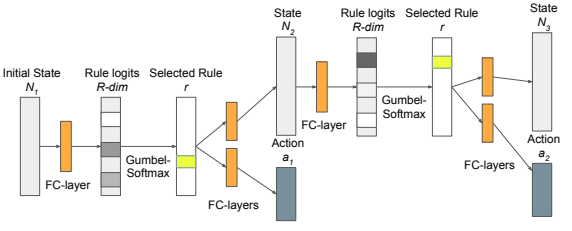

Figure 2: Overview of the stochastic recurrent model which generates an output action per step and a latent state (which will in turn generate next actions). Each time the model is run, a different rule is selected, thanks to the Gumbel-Softmax trick, leading to a different action and state. This results in multiple sequences (see text for more details).

The central idea is to learn a stochastic model that generates multiple, different candidate sequences, which can be ranked based on instructional video constraints. The top ranked sequence is used as self-labels to train the action segmentation model (Figure 1). By iterating this process in an EM-like procedure, the model converges to a good segmentation of actions.

We evaluate the approach on multiple datasets and compare to previous methods on unsupervised action segmentation. We also compare to weakly-supervised and supervised baselines. Our unsupervised method outperforms all state-of-the-art models, in some cases considerably, with performance at times outperforming weakly-supervised methods.

Our contributions are (1) a stochastic model capable of capturing multiple possible sequences, (2) a training method that is able to learn to segment actions without any labeled data, (3) a new state-of-the art in unsupervised segmentation for instructional videos.

2 Related Work

Studying instructional videos has gained a lot of interest recently [1, 6, 21, 34], largely fueled by advancements in feature learning and activity recognition for videos [5, 8, 26, 35, 36]. However, most work on activity segmentation has focused on the fully-supervised case [29, 38], which requires per-frame labels of the occurring activities.

Since it is expensive to fully annotate videos, weakly-supervised activity segmentation has been proposed. Initial works use movie scripts to obtain weak estimates of actions

[17, 20] or localize actions based on related web images [9, 10, 33]. [3] perform weakly-supervised segmentation when assuming the ordering was given, both during training and test time. Temporal ordering constraints [12] or language [2, 28, 37] have also been applied to learn segmentation. Related 'set-supervised' learning [7, 18, 24] only assumes the actions in the video are known, but not the ordering.

Several unsupervised methods have also been proposed [1, 16, 27, 31]. Alayrac et al. [1] learn action segmentation without segmentation supervision, using text in addition to video data. [16] uses k-means clustering to do a time-based clustering of features and the Viterbi algorithm segment the videos based on the clusters. [27] uses a GMM to learn a transition model between actions. We propose a fully differentiable unsupervised action segmentation, which works from RGB inputs only.

Several datasets for learning from instructional videos have been introduced recently: Breakfast [14], 50-salads [32], the Narrated Instructional Videos (NIV) [1], COIN [34], HowTo100m [21], CrossTask [39] and PROCEL [6].

3 Method

Our goal is to discover atomic actions from a set of instructional videos, while capturing and modeling their temporal structure. Formally, given a set of videos $\mathcal{V} = \{V^1, V^2, ...\}$ of a task or set of tasks, the objective is to learn a model that maps a sequence of frames $V^i = [I_t]_{t=1}^T$ from any video to a sequence of atomic action symbols $[a_t \in \mathcal{O}]_{t=1}^T$ where $\mathcal{O}$ is a set of possible action symbols (we drop the index i for simplicity).

Supervised approaches assume that each frame is labeled with an action, and most weakly supervised approaches assume the actions per video are given in their correct order, but without start and end times. In the unsupervised case, similar to previous works [1, 16], we assume no action labels or boundaries are given. To evaluate the approach, we follow the previous setting using the Hungarian algorithm to match predicted actions to ground truth labels. While previous methods used additional data such a subtitles or text [1], the proposed approach does not use such information. Our model, however, works with a fixed k-the number of actions per task (analogous to setting k in k-means clustering), and we run it with a range of values for k. This is not a very strict assumption as the number of expected atomic actions per instruction is roughly known, e.g., about 10 actions for doing CPR, or 40 actions when making a salad. For example, a video of making a fried egg will contain the same atomic actions: e.g., cracking the egg, heating a pan, frying the egg, and serving. However, the temporal order, duration and appearance of the actions will vary across videos.

3.1 Sequential Stochastic Autoregressive Model

Our method is based on a sequential stochastic autoregressive model (e.g., [4, 22]). The model consists of three components: $(\mathcal{H}, \mathcal{O}, \mathcal{R})$ where $\mathcal{H}$ is a finite set of states, $\mathcal{O}$ is a finite set of output symbols, and $\mathcal{R}$ is a finite set of transition rules mapping from a state to an output symbol and next state. Importantly, this model is stochastic, i.e., each rule is additionally associated with a probability of being selected, and thus the sum of the rule probabilities for a given state is 1. Note that during training, $\mathcal{O}$ is just a set of symbols with no semantic meaning or connection to the ground truth labels. For evaluation, following previous works ([16]), we use the Hungarian algorithm to match these to ground truth symbols.

To implement this method in a differentiable way, we use fully-connected layers and the Gumbel-Softmax trick [13, 19]. Specifically, we use several FC layers taking the current state as input and outputting a vector of logits, representing probabilities of each rule being selected. Next, using the Gumbel-Softmax trick, the model differentiably samples one of the rules. Each time this function is run, a different rule can be selected, learning to generate different sequences (Figure 2). This property is important for the learning of dependencies in sequences.

For a full video $V = [I_1, I_2, I_3, \ldots, I_T]$ as input, where each I_t is an RGB image frame from the video, we process the frames by some CNN (e.g., ResNet, I3D [5], AssembleNet [26], we use the latter), resulting in a sequence of feature vectors, $[f_1, f_2, \ldots, f_T]$. These features are used as input to the model, which will generate a sequence of output symbols $S = [a_1, a_2, \ldots, a_T]$ as follows:

$$\begin{aligned} a_1, N_1 &= G(N_0, f_1), \\ a_2, N_2 &= G(N_1, f_2), \\ a_T, N_T &= G(N_{T-1}, f_T) \end{aligned} \tag{1}$$

The model takes each feature as input and concatenates it with the state which is used as input to G to produce the output. Once applied to every frame, this results in a sequence of actions. We note that the size of $\mathcal{O}$, k, is a hyper-parameter and controls the number of atomic actions expected in the videos.

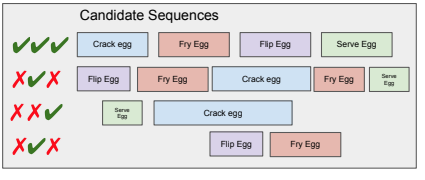

Figure 3: Multiple candidate sequences are generated and ranked. The best sequence according to the ranking function is chosen as the labels for the iteration.

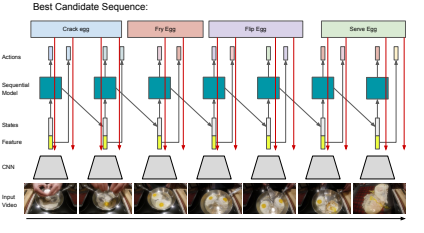

Figure 4: Once the best candidate sequence is selected, it is used to train the model using standard backpropagation. Both the state sequence model as well as the FC-layers generating frame predictions are trained.

3.2 Learning by Self-Labeling of Videos

In order to train the model without ground truth action sequences, we introduce an approach of learning by 'self-labeling' videos. The idea is to optimize the model by generating self-supervisory labels that best satisfies the constraints required for atomic actions. Notably, the stochastic ability to generate multiple sequences is key to this approach. As a result of the learning, a sequence with better constraint score will become more likely to be generated than the sequences with worse scores.

We first generate multiple candidate sequences, then rank them based on the instructional video constraints, which importantly require no labeled data. Since the Gumbel-Softmax adds randomness to the model, the output can be different each time G is run with the same

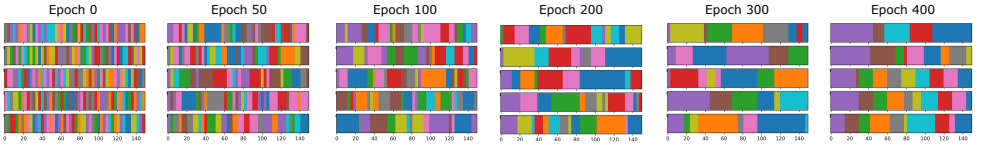

Figure 5: Candidate sequences at different stages of training. The sequences shown are the top 5 ranked sequences (rows) at the given epoch. The top one is selected as supervision for the given step. The colors represent the discovered action (with no labels).

Method	F1 score
Supervised Baselines	
VGG [30], from Alayrac et al. [1]	0.376
I3D, Carreira et al. [3]	0.472
AssembleNet, Ryoo et al. [26]	0.558
Weakly-supervised	
CTC, Huang et al. [12] +AssembleNet [26]	0.312
ECTC, Huang et al. [12] +AssembleNet [26]	0.334
Unsupervised	
Uniform Sampling	0.187
Alayrac et al. [1]	0.238
Kukleva et al [16]	0.283
JointSeqFL, Elhamifar et al. [3]	0.373
Ours	**0.457**

Table 1: Results on the NIV dataset

input, which is key to the approach. Specifically, the model is run M times, giving M potentially different sequences of actions. We then define a cost function to rank each of the M sequences. The top ranked sequence is selected as the labels which are used for learning. This ranking function constrains the possible generated sequences. The ranking function we propose to capture the structure of instructional videos has multiple components: (1) Every atomic action must occur once in the task. (2) Every atomic action should have similar lengths across videos of the same task. (3) Each symbol should reasonably match the provided visual feature (Figure 3). They are all modeled by losses in our approach, as shown below, and can be relaxed if needed, e.g. if more than one occurrence is expected:

Action Occurrence: Given a sequence S of output symbols (i.e., actions), the first constraint ensures that every action appears once. Formally, it is implemented as $C_1(S) = |\mathcal{O}| - \sum_{a \in \mathcal{O}} \text{App}(a)$, where App is 1 if a appears in S otherwise it is 0. This constraint is optional, but we include it as it is a property of instructional videos that can be leveraged.

Modeling Action Length: The constraint ensuring each atomic action has a similar duration across different videos can be implemented in several different ways. The simplest approach is to compute the difference in length compared to the average action length in the video (the exact equation is in the appendix).

Another way to model length is by considering the duration of an action to be drawn from a distribution (e.g., Poisson or Gaussian).

$$C_2(S) = \sum_{a \in \mathcal{O}} (1 - p(L(a,S))), \tag{2}$$

Modeling Action Probability: The third constraint is implemented using the separate classification layer of the network $p(a|f)$, which gives the probability of the frame being

Method	MoF
Supervised Baselines	
VGG [30] [1]	60.8
I3D [5]	72.8
AssembleNet [26]	77.6
Weakly-supervised	
CTC [12]	11.9
HTK [15]	24.7
HMM + RNN [23]	45.5
NN-Viterbi [25]	49.4
Ours, weakly supervised [1]	53.7
Unsupervised	
Kukleva et al [16]	30.2
Ours	**39.7**

Table 2: Results on the 50-salads dataset.

Method	MoF	Jaccard
Supervised Baselines		
VGG [30], [1]	62.8	75.4
I3D, [5]	67.8	79.4
AssembleNet, [26]	72.5	82.1
Weakly-supervised		
OCDC, [12]	8.9	23.4
ECTC, [12]	27.7	-
HMM + RNN, [23]	33.3	47.3
Unsupervised		
SCV, [18]	30.2	-
SCT, [2]	30.4	-
Sener et al [27]	34.6	47.1
Kukleva et al [16]	41.8	-
Ours	**43.5**	**54.4**

Table 3: Results on the Breakfast dataset.

classified as action a. Formally, $C_3(S) = \sum_{t=1}^{T}(1 - p(a_t|f_t))$, which is the probability that the given frame belongs to the selected action. This constraint is separate from the sequential model and captures independent appearance based probabilities.

We can then compute the rank of any sequence as $C(S) = \gamma_1 C_1(S) + \gamma_2 C_2(S) + \gamma_3 C_3(S)$, where γ_i weights the impact of each term. In practice setting γ_2 and γ_3 to $\frac{1}{|S|}$ and $\gamma_1 = \frac{1}{|\mathcal{O}|}$ works well.

Learning Actions: To choose the self-labeling, we sample K sequences, compute each cost and select the sequence that minimizes the above cost function. This gives the best segmentation of actions (at this iteration of labeling), based on the defined constraints.

$$\hat{S} = \texttt{argmin}_S C(S). \tag{3}$$

We note that this cost function does not need to be differentiable. The cost function is only used to choose the self-labels. Once the labels are selected, the standard cross-entropy loss function with backpropagation is used to train the model. The cost function gives a strong prior for how to choose the labels without any annotations, and allows unsupervised learning.

$$\mathcal{L}(\hat{S}, A, P) = -\sum_{i \in \mathcal{O}} \sum_{t=1}^{T} \hat{a}_{t,i} \log(a_{t,i}) + \hat{a}_{t,i} \log(p_{t,i}). \tag{4}$$

This loss trains both the classification layer as well as the model.

We also allow a null class to indicate that no actions are occurring in the given frames. This class is not used in any of the above constraints, i.e., it can occur wherever it wants, for as long as needed and as many times as needed. We omit frames labeled with the null class when calculating the cost function and find that the constraint encouraging each action to occur once eliminates the solution where only the null class is chosen.

Cross-Video Matching: The above constraints work reasonably well for a single video, however when we have multiple videos with the same actions, we can further improve the ranking function by adding a cross-video matching constraint. The motivation for this is that while breaking an egg can be visually different between two videos, the actions are the same. Given a video segment the model labeled as an action f_a from one video, a segment

Method	NIV	50-Salads	Brkfst
Supervised	0.558	77.6	72.5
CTC	0.312	42.8	38.7
RNN + CTC	0.388	47.9	42.4
Ours + CTC	**0.480**	**52.8**	**45.3**

Table 4: Comparing different weakly-supervised models. All using AssembleNet features. The supervised counterpart at the top.

Cost	50-Salads	Brkfst
Randomly pick candidate	12.5	10.8
No Gumbel-Softmax	10.5	9.7
Occurrence (C_1)	22.4	19.8
Length (C_2)	19.6	17.8
$p(a\|f)$ (C_3)	21.5	18.8
C_1+C_2	27.5	25.4
C_1+C_3	30.3	28.4
C_2+C_3	29.7	27.8
$C_1+C_2+C_3$	33.4	29.8

Table 5: Ablation with cost function terms[2]

$\hat{f}_a$ the model labeled as the same action from a second video, and a segment f_b the modeled labeled as a different action from any video, we can measure the cross-video similarity using a triplet loss

$$\mathcal{L}_T(f_a, \hat{f}_a, f_b) = ||f_a - \hat{f}_a||_2 - ||f_a - f_b||_2 + \alpha, \tag{5}$$

or a contrastive loss

$$\mathcal{L}_C(f_a, \hat{f}_a, f_b) = \frac{1}{2}||f_a - \hat{f}_a||_2 + \frac{1}{2}\max(0, \alpha - ||f_a - f_b||_2). \tag{6}$$

As these functions are differentiable, we can directly add this to the loss function (Eq. 4) or to the cost function (e.g. as an additional cost term similar to C_2 in Eq. 2) or both. By adding this to the cost function, we are ensuring that the chosen labeling of the videos is most consistent for feature representations. By adding it to the loss function, we are encouraging the learned representations to be similar for the actions with the same selected labels and different for other actions. We experimentally compare these in the supplemental material.

3.3 Self-labeling Training Method

Using the previous components, we now describe the full training method, which follows an EM-like procedure. In the first step, we find the optimal set of action self-labels, given the current model parameters and the ranking function. In the second step, we optimize the model parameters, and optionally some ranking function parameters, for the selected self-labeling (Figure 4). After taking both steps, we have completed one iteration. Following standard neural network training, we do each step for a mini-batch of 32 samples. The model is trained for 500 epochs. Due to the iterative update of the labels at each step, we observe that this method requires more epochs than supervised learning.

Learning action length: As an optional training phase, we update some parameters of the ranking function. The particular parameters to learn are those determining the length of each action, since some atomic actions will be longer than others and we often do not know the actual length of the action. To do this, we modify the length model so that it has a λ_a or μ_a, σ_a to represent the length of each action a. To estimate these values, after the backpropagation of the gradients, we run the model in inference mode to obtain a segmentation of the video. For each action, we then compute its average length (and optionally variance) which we can use to update λ_a or μ_a, σ_a.

Segmenting a video at inference: CNN features are computed for each frame and the model is applied on those features. While running the SAM, we greedily select the most

Method	chng tire	CPR	repot plant	make coffee	jump car	Avg.
Alaryac et al. [1]	0.41	0.32	0.18	0.20	0.08	0.238
Kukleva et al. [16]	-	-	-	-	-	0.283
Ours VGG	0.53	0.46	0.29	0.35	0.25	0.376
Ours AssembleNet	0.63	0.54	0.381	0.42	0.315	0.457

Table 6: Comparison on the NIV dataset of the proposed approach on VGG and AssembleNet features. Our approach outperforms others, even when using weaker VGG features.

probable rule to generate the action labels. Future work can improve this by considering multiple possible sequences (e.g., following the Viterbi algorithm).

4 Experiments

We evaluate our unsupervised atomic action discovery approach on multiple video segmentation datasets, establishing a new state-of-the-art for unsupervised segmentation. Our qualitative results confirm too that the self-generated action annotations form meaningful action segments. We note that there is only a handful of methods that have attempted unsupervised activity segmentation. More results can be seen in the supplemental materials. Thus, we also compare to several fully-supervised methods and to weakly-supervised ones.

Datasets: We compare results on the **50-salads dataset** [32]. The videos contain the same set of actions (e.g., cut lettuce, cut tomato, etc), but the ordering of actions is different in each video. We compare on **Narrated Instructional Videos (NIV) dataset** [1], which contains 5 different tasks. Finally, we use the **Breakfast dataset** [14] which contains videos of people making breakfast dishes from various camera angles and environments. We chose these datasets as they cover a wide variety of approaches focused on unsupervised, weakly-supervised, and fully-supervised action segmentation, allowing a comparison to them.

Evaluation Metrics: We follow all previously established protocols for evaluation in each dataset. We first use the Hungarian algorithm to map the predicted action symbols to action classes in the ground truth. Since different metrics are used for different datasets we report the previously adopted metrics per dataset.

4.1 Comparison to the state-of-the-art

In Tables 1, 2, 3, we compare our approach to previous state-of-the-art methods. While there are few works on the fully unsupervised case, we note that our approach, together with strong video features, provides better segmentation results than previous unsupervised and even weakly-supervised methods (JointSeqFL [6] uses optical flow and does not provide results on 50-salads or Breakfast).

For full comparison we include strong supervised baselines, e.g., I3D [5], and AssembleNet [26]. We also use implementations of the CTC [11] and ECTC [12] methods using the AssembleNet backbone [26]. Our unsupervised approach outperforms many weakly-supervised ones too (Tables 1, 3).

Qualitative Analysis In Figure 5, we show the generated candidate sequences at different stages of learning. Initially, the generated sequences are entirely random and over-segmented. As training progresses, the generated sequences start to match the constraints.

[1] For the weakly-supervised setting, we use activity order as supervision, equivalent to previous works.

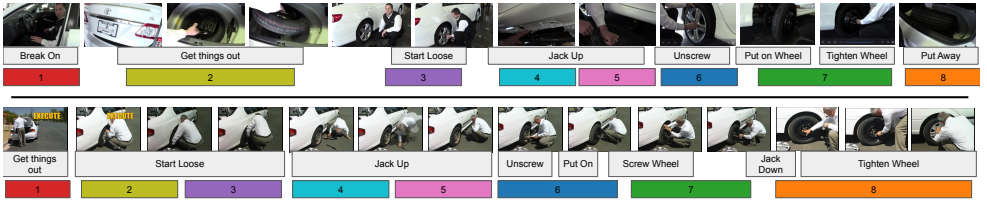

Figure 6: Two example videos from the 'change tire' activity. The ground truth is shown in grey, the model's top rank segmentation is shown in colors. NIV dataset.

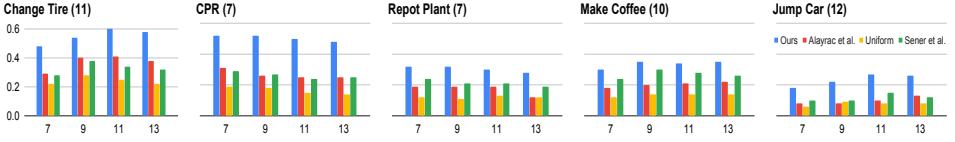

Figure 7: F1 value for varying the number of actions used, compared to prior work. The number in parenthesis indicates the ground-truth number of actions for each activity.

After 400 epochs, the generated sequences show similar order and length constraints, and better match the ground truth (as shown in the evaluation). Figure 6 shows example results of our method.

4.2 Ablation experiments

Effect of sequential models for weak-supervision. We conduct a set of experiments to determine the effect of learning temporal information in different ways. The CTC loss, which bases the loss on the probability of the sequence occurring [□], can be applied directly on per-frame features, without any underlying RNN or temporal model. In Table 4, we compare the effect of using the CTC loss with per-frame CNN features, an RNN, and our model.

Effects of the cost function constraints. To determine how each cost function impacts the resulting performance, we compare various combinations of the terms. The results are shown in Table 5. We find that each term is important to the self-labeling of the videos[2]. Generating better self-labels improves model performance, and each component is beneficial to the selection process. We also compare to other baselines.

Methods for cross-video matching. In the supplemental material, we compare the results for the different methods of cross-video matching on the 50-salads dataset. We find that using the contrastive as part of the training loss performs the best, as this further encourages the learned representation to match the chosen labels.

Methods for length modeling. In the supplemental material, we compare the different methods to model the length of each action. We find that learning the length of each action is most beneficial.

Varying the number of actions. As $\mathcal{O}$ is a hyper-parameter controlling the number of actions to segment the video into, we conduct experiments varying the number of actions to evaluate the effect of this hyper-parameter. The results are shown in Figure 7. We find that the model is not overly-sensitive to this hyper-parameter, but it does have some impact on the performance due to the fact that each action must appear at least once in the video.

[2]These ablation methods do not use our full cross-video matching or action duration learning, thus the performances are slightly lower than the our best results.

Feature comparisons. As our work uses AssembleNet [26] features, in Table 6 we compare the proposed approach to previous ones using both VGG and AssembleNet features. As shown, even using VGG features, our approach outperforms previous methods.

Conclusion

We present a novel approach for unsupervised action segmentation for instructional videos, which learns to effectively self-label and learn a segmentation of actions. Our work demonstrates strong results and outperforms the state-of-the-art. Code will be open sourced.

References

[1] 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 (CVPR)*, pages 4575–4583, 2016.

[2] Lisa Anne Hendricks, Oliver Wang, Eli Shechtman, Josef Sivic, Trevor Darrell, and Bryan Russell. Localizing moments in video with natural language. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 5803–5812, 2017.

[3] 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 *Proceedings of European Conference on Computer Vision (ECCV)*, pages 628–643. Springer, 2014.

[4] Andrew D Brown and Geoffrey E Hinton. Products of hidden markov models. In *AISTATS*. Citeseer, 2001.

[5] 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 (CVPR)*, 2017.

[6] Ehsan Elhamifar and Zwe Naing. Unsupervised procedure learning via joint dynamic summarization. 2019.

[7] Mohsen Fayyaz and Jurgen Gall. Sct: Set constrained temporal transformer for set supervised action segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[8] Christoph Feichtenhofer, Haoqi Fan, Jitendra Malik, and Kaiming He. Slowfast networks for video recognition. *arXiv preprint arXiv:1812.03982*, 2018.

[9] Chuang Gan, Chen Sun, Lixin Duan, and Boqing Gong. Webly-supervised video recognition by mutually voting for relevant web images and web video frames. In *Proceedings of European Conference on Computer Vision (ECCV)*, pages 849–866. Springer, 2016.

[10] Chuang Gan, Ting Yao, Kuiyuan Yang, Yi Yang, and Tao Mei. You lead, we exceed: Labor-free video concept learning by jointly exploiting web videos and images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 923–932, 2016.

[11] Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In *International Conference on Machine Learning (ICML)*, pages 369–376, 2006.

[12] De-An Huang, Li Fei-Fei, and Juan Carlos Niebles. Connectionist temporal modeling for weakly supervised action labeling. In *Proceedings of European Conference on Computer Vision (ECCV)*, 2016.

[13] Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. In *International Conference on Learning Representations*, 2017.

[14] Hilde Kuehne, Ali Arslan, and Thomas Serre. The language of actions: Recovering the syntax and semantics of goal-directed human activities. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 780–787, 2014.

[15] Hilde Kuehne, Alexander Richard, and Juergen Gall. Weakly supervised learning of actions from transcripts. *Computer Vision and Image Understanding (CVIU)*, 2017.

[16] Anna Kukleva, Hilde Kuehne, Fadime Sener, and Jurgen Gall. Unsupervised learning of action classes with continuous temporal embedding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12066–12074, 2019.

[17] Ivan Laptev, Marcin Marszalek, Cordelia Schmid, and Benjamin Rozenfeld. Learning realistic human actions from movies. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8. IEEE, 2008.

[18] Jun Li and Sinisa Todorovic. Set-constrained viterbi for set-supervised action segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[19] Chris J Maddison, Andriy Mnih, and Yee Whye Teh. The concrete distribution: A continuous relaxation of discrete random variables. In *International Conference on Learning Representations*, 2017.

[20] Marcin Marszalek, Ivan Laptev, and Cordelia Schmid. Actions in context. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2929–2936. IEEE, 2009.

[21] 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. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2630–2640, 2019.

[22] AJ Piergiovanni, Anelia Angelova, and Michael S Ryoo. Differentiable grammars for videos. In *Proceedings of AAAI Conference on Artificial Intelligence (AAAI)*, 2020.

[23] Alexander Richard, Hilde Kuehne, and Juergen Gall. Weakly supervised action learning with rnn based fine-to-coarse modeling. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[24] Alexander Richard, Hilde Kuehne, and Juergen Gall. Action sets: Weakly supervised action segmentation without ordering constraints. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5987–5996, 2018.

[25] Alexander Richard, Hilde Kuehne, Ahsan Iqbal, and Juergen Gall. NeuralNetwork-Viterbi: A framework for weakly supervised video learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7386–7395, 2018.

[26] Michael S. Ryoo, AJ Piergiovanni, Mingxing Tan, and Anelia Angelova. Assemblenet: Searching for multi-stream neural connectivity in video architectures. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=SJgMK64Ywr.

[27] Fadime Sener and Angela Yao. Unsupervised learning and segmentation of complex activities from video. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8368–8376, 2018.

[28] Ozan Sener, Amir R Zamir, Silvio Savarese, and Ashutosh Saxena. Unsupervised semantic parsing of video collections. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 4480–4488, 2015.

[29] Zheng Shou, Jonathan Chan, Alireza Zareian, Kazuyuki Miyazawa, and Shih-Fu Chang. Cdc: Convolutional-de-convolutional networks for precise temporal action localization in untrimmed videos. *arXiv preprint arXiv:1703.01515*, 2017.

[30] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[31] Khurram Soomro and Mubarak Shah. Unsupervised action discovery and localization in videos. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2017.

[32] Sebastian Stein and Stephen J McKenna. Combining embedded accelerometers with computer vision for recognizing food preparation activities. In *Proceedings of the 2013 ACM international joint conference on Pervasive and ubiquitous computing*, pages 729–738, 2013.

[33] Chen Sun, Sanketh Shetty, Rahul Sukthankar, and Ram Nevatia. Temporal localization of fine-grained actions in videos by domain transfer from web images. In *Proceedings of the 23rd ACM international conference on Multimedia*, pages 371–380, 2015.

[34] 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 *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1207–1216, 2019.

[35] Du Tran, Heng Wang, Lorenzo Torresani, Jamie Ray, Yann LeCun, and Manohar Paluri. A closer look at spatiotemporal convolutions for action recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6450–6459, 2018.

[36] Saining Xie, Chen Sun, Jonathan Huang, Zhuowen Tu, and Kevin Murphy. Rethinking spatiotemporal feature learning: Speed-accuracy trade-offs in video classification. In *Proceedings of European Conference on Computer Vision (ECCV)*, pages 305–321, 2018.

[37] Luowei Zhou, Chenliang Xu, and Jason J Corso. Towards automatic learning of procedures from web instructional videos. In *Proceedings of AAAI Conference on Artificial Intelligence (AAAI)*, 2018.

[38] Hongyuan Zhu, Romain Vial, and Shijian Lu. Tornado: A spatio-temporal convolutional regression network for video action proposal. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5813–5821, 2017.

[39] D. Zhukov, J.-B. Alayrac, R. G. Cinbis, D. Fouhey, and. Laptev, and andJ. Sivic. Cross-task weakly supervised learning from instructional videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.