Self-Supervised Multi-Task Procedure Learning from Instructional Videos

Ehsan Elhamifar[0000−0001−5517−548X] and Dat Huynh[0000−0002−6349−2597]

Khoury College of Computer Sciences, Northeastern University, Boston, USA
{e.elhamifar,huynh.dat}@northeastern.edu

Abstract. We address the problem of unsupervised procedure learning from instructional videos of multiple tasks using Deep Neural Networks (DNNs). Unlike existing works, we assume that training videos come from multiple tasks without key-step annotations or grammars, and the goals are to classify a test video to the underlying task and to localize its key-steps. Our DNN learns task-dependent attention features from informative regions of each frame without ground-truth bounding boxes and learns to discover and localize key-steps without key-step annotations by using an unsupervised subset selection module as a teacher. It also learns to classify an input video using the discovered key-steps using a learnable key-step feature pooling mechanism that extracts and learns to combine key-step based features for task recognition. By experiments on two instructional video datasets, we show the effectiveness of our method for unsupervised localization of procedure steps and video classification.

Keywords: procedure learning, instructional videos, subset selection, self-supervised learning, deep neural networks, attention modeling

1 Introduction

The large number of instructional and everyday activity videos has provided great resources for automatic procedure learning (APL), which is to learn the sequence and visual models of key-steps required to achieve a certain task. Procedure learning can be used to teach autonomous agents perform complex tasks [33], help humans in achieving tasks [28], or build large knowledge bases of instructions. Understanding videos at the scale necessary to build knowledge bases or assistive robots that handle a large number of tasks requires unsupervised methods that do not rely on costly to gather annotated videos.

1.1 Prior Work

Over the past few years, we have seen advances on multiple aspects of understanding instructions [9, 31, 29, 1, 27, 12, 18, 22, 36, 20]. Depending on the type of supervision, existing works can be divided into three categories. The first group of works assumes that annotations of key-steps (also referred to as procedure steps) are given in videos and the goal is to learn how to segment new videos

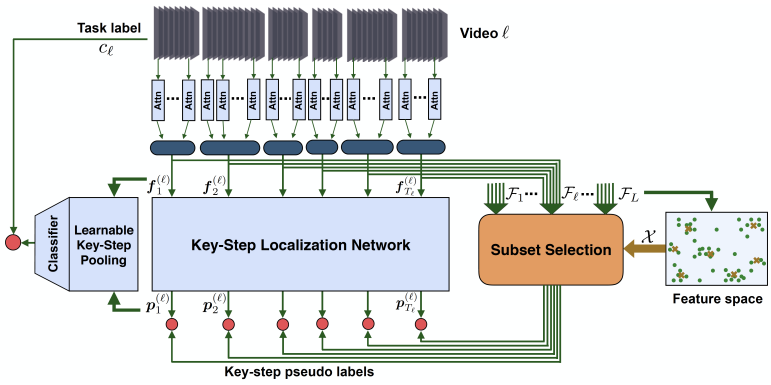

Fig. 1: We develop a self-supervised method for key-step localization and task classification using a deep network. Our framework uses unsupervised subset selection to self-supervise training of a key-step localization network (KLN), where both share attention features that focus on task-related informative regions in video frames. The attention features and output of KLN will be used in a task-classification network (TCN) that uses a learnable key-step pooling mechanism to automatically upweight discriminative key-steps.

[36] or anticipate future key-steps [28]. To reduce the costly and unscalable annotation requirement, weakly supervised learning methods [13, 23, 3, 5, 37] assume that each video is accompanied with an ordered or unordered list of key-steps (subactions) appearing in it, and the goals are to localize the key-steps in videos and learn a model for each key-step. [31] further reduces the annotation cost by only marking one frame from each key-step in videos without requiring to label the selected frames. While removing the stringent requirement of annotating each frame, the weakly supervised methods still require annotators to watch each video entirely and provide its (ordered) list of key-steps.

To remove the need for annotation, unsupervised procedure learning methods have focused on exploiting the structure of videos of the same task in order to discover and localize key-steps in videos [29, 1, 27, 9, 16, 10]. Several works have addressed understanding procedures from narration or text [29, 1, 18, 34, 6]. However, reliably obtaining text from spoken natural language using videos on the Internet is still challenging, often requiring manual cleaning of the automatic speech recognition results. Moreover, to learn visual models of key-steps, existing methods assume that the text and visual information are aligned [1, 18, 34], which could be violated in videos, e.g., human narrators first speak about one or multiple key-steps and then perform the subactions. Thus, to learn reliable visual models of key-steps, recent works have focused on learning key-steps directly from visual data [27, 9, 16], using a Mallows model [27], joint dynamic summarization [9] or clustering and ordering of visual features [16].

Limitations. Existing works on unsupervised procedure learning are limited in three aspects. First, most methods assume videos of only one task are given,

with the goal of discovering a common procedure as well as the localization of key-steps in videos [1, 27, 9, 16]. This requires running such methods on each task separately, while it is not clear how to discover the key-steps of a new video that may belong to any of the tasks.

On the other hand, despite great success of DNNs for recognition, detection, captioning, semantic segmentation and tracking, they have not been fully explored for procedure learning. This comes from the difficulty of the unsupervised problem, which often requires a multi-stage solution with each stage involving a non-convex problem, e.g., multiple sequence alignment and discriminative clustering [1], temporal embedding, clustering and decoding [16] or subset selection and multiple sequence alignment [9]. Thus, unsupervised procedure learning based on DNNs, which once learned allows fast and efficient inference on a test video belonging to one of many possible tasks, remains an important yet challenging problem to address.

Finally, the majority of existing works on unsupervised procedure learning lack the ability to learn informative features for key-step discovery, often relying on precomputed features. Recent works have focused on using narration and visual data jointly to learn features [20, 19], yet they require access to narrations and rely on weak alignment between the modalities. In this paper, we focus on *visual data only*, motivated by problems such as learning from home activity videos where users do not explain their actions while performing different tasks. We make the key observation that in instructional videos, information about a key-step is often contained in a small region of a frame (e.g., for the key-step 'unscrew the wheel' in the 'change tire' task, the information is contained in the region of a frame that contains the lug nuts). Hence, to effectively perform procedure learning, we need to successfully localize and extract features from regions in each frame where a key-step is being performed.

1.2 Paper Contributions

We develop a self-supervised procedure learning method from videos of multiple tasks using Deep Neural Network (DNNs), addressing the above challenges. Unlike existing works that assume all videos come from the same task, we consider the more challenging and practical scenario where training videos come from different tasks without key-step annotations or grammars, and the goals are to classify a test video to the underlying task and to localize its key-steps. To tackle the problem, we study a DNN-based framework (see Figure 1) that

- learns *task-dependent attention features* from informative regions of frames using a self-attention mechanism trained without bounding box annotations;
- learns to discover and *localize key-steps without ground-truth annotations of key-steps* by using an *unsupervised subset selection* module as a teacher;
- learns to *classify a video using a learnable key-step feature pooling mechanism* that extracts and learns to combine key-step based features.

While subset selection does not allow backpropagation of gradients due to its discrete non-differentiable nature, our framework allows us to *learn attention*

features for both DNN and subset selection, hence, improving the performance of both for key-step localization and task classification. In addition, the unsupervised subset selection provides pseudo labels for DNN to learn to localize key-steps in videos. Our experiments show that once learned, the DNN outperforms subset selection in key-step localization, thanks to its higher capacity.

Remark 1. Our framework can be thought as a *teacher-student* framework, with major differences with most existing works [2, 11, 25, 35, 21]: i) our teacher is a non-convex subset selection, which unlike prior work is not first learned and then fixed; ii) both the teacher (subset selection) and the student (key-step localization network) share the same attention feature learning module, where the supervision by the non-differentiable teacher and back-propagating gradient of loss via the differentiable student, allows to learn better (attention) features that subsequently *improve the performance of both.*[1]

2 Self-Supervised Procedure Learning

Assume we have a training set of videos and their labels $\{(\mathcal{Y}_\ell, c_\ell)\}_{\ell=1}^{L}$, where $\mathcal{Y}_\ell = (\boldsymbol{y}_1^{(\ell)}, \ldots, \boldsymbol{y}_{T_\ell}^{(\ell)})$ denotes the sequence of feature vectors of the video ℓ with T_ℓ frames and $\boldsymbol{y}_t^{(\ell)}$ denotes the feature vector of the t-th frame. Also, $c_\ell \in \mathcal{A}$ is the task label of the video ℓ, where $\mathcal{A}$ is the set of labels of all tasks, e.g., $\mathcal{A} =$ {'change tire', 'jump-start car', ...}. Given a desired procedure length k, which is a hyperparameter, our goal is to recover in a test video all frames that belong to each of the k key-steps and the underlying task label of the video.

2.1 Proposed Framework

We develop a DNN-based method for efficient key-step discovery and video classification by learning from training videos that do not have key-step annotations or grammars. To do so, we propose a framework in which the DNN component that predicts assignments to key-steps is self-supervised by the output of an unsupervised subset selection module on training videos, while the DNN component that predicts the task label is supervised by the ground-truth task labels of training videos. Moreover, the DNN and unsupervised subset selection use the same attention features as inputs. This allows us to backpropagate gradients for the attention module via DNN. More specifically, Our framework consist of the following components:

– A *spatial attention network (SAN)* that learns to focus on informative task-related regions of each frame to extract feature from. The attention mechanism in our work does not require any ground-truth bounding boxes and is learned with self-supervision on key-steps and supervision on task labels. As we show in the experiments, the presence of this component significantly improves the performance of both DNN and subset selection.

[1] A similar idea was used in [4] to train a deep classification network using kmeans.

– An *unsupervised subset selection* component whose inputs are the attention features of all videos and M latent states learned from features that ideally capture different key-steps across tasks and background subactions. It then selects a subset of the latent states as key-steps and finds assignments of video frames to them, hence localizing key-steps. We use the output of the subset selection as pseudo labels for training the DNN.
– A *key-step localization network (KLN)* that receives the attention feature of each frame in a video and outputs an M-dimensional vector for each frame that corresponds to the probability vector of the frame belonging to each of the M states. We use the output of the subset selection to provide supervisory information, i.e., pseudo labels, for key-step localization in KLN.
– A *task classification network (TCN) with learnable key-step feature pooling* that uses the attention features and key-step assignment probabilities (i.e., input and output of KLN) and predicts the video task label. Our key observation is that some key-steps (e.g., the ones common across tasks) should be less emphasized for video classification, while discriminative key-steps (e.g., the ones more specific to a particular task) should be more emphasized. Thus, we propose to build M key-step based features and a learnable pooling mechanism that learns to combine and attend to discriminative key-step based features.

2.1.1 Spatial Attention Network (SAN). We make the observation that task-related discriminative information about each key-step is contained in specific regions of a frame, e.g., in the task of 'changing tire', for the key-step of 'loosen lug nuts', the information is contained in the region of the frame that contains the lug nuts and the wheel. Thus, we use a spatial attention module [32, 15, 14] that, without costly ground-truth bounding box supervision, learns to extract features from the most informative regions of a frame.

We divide the frame t of the video ℓ, denoted by $I_t^{(\ell)}$, into R equal-sized grid cells, denoted by $I_{t,1}^{(\ell)}, \ldots, I_{t,R}^{(\ell)}$. Let $\boldsymbol{y}_{t,r}^{(\ell)} = f_\Theta(I_{t,r}^{(\ell)})$ be the feature vector of the region r, extracted using a CNN parametrized by Θ (see experiments for the details). Given region features $\{\boldsymbol{y}_{t,r}^{(\ell)}\}_{r=1}^{R}$, the spatial attention module learns to find the most relevant regions to the underlying key-step. This is done by finding an attention feature, $\boldsymbol{f}_t^{(\ell)}$, defined as

$$\boldsymbol{f}_t^{(\ell)} = \sum_{r=1}^{R} \alpha_r\big(\boldsymbol{y}_{t,r}^{(\ell)}\big)\boldsymbol{y}_{t,r}^{(\ell)}, \quad \alpha_r\big(\boldsymbol{y}_{t,r}^{(\ell)}\big) = \frac{\exp(m_\alpha(\boldsymbol{y}_{t,r}^{(\ell)}))}{\sum_{r'=1}^{R}\exp(m_\alpha(\boldsymbol{y}_{t,r'}^{(\ell)}))} \tag{1}$$

where $\alpha_r\big(\boldsymbol{y}_{t,r}^{(\ell)}\big)$ denotes the weight of selecting the region r. These weights are computed by a neural network whose outputs $m_\alpha(\boldsymbol{y}_{t,r}^{(\ell)})$ for given inputs $\{\boldsymbol{y}_{t,r}^{(\ell)}\}_{r=1}^{R}$ are normalized by a softmax function across all image regions. The weights are unknown and the task of the attention module is to find them for an input image.

2.1.2. Subset Selection. The goal of subset selection is to find a small subset of representative points from a dataset and it has shown promising results for

procedure learning [9, 31, 7, 8]. Here, we use unsupervised subset selection to provide pseudo key-step labels for the DNN. More specifically, the subset selection takes attention features of all videos and M learned latent states (corresponding to key-steps and background subactions across tasks) and selects a subset of the states as key-steps and finds the assignments of the frames to them.

Given $\{\boldsymbol{f}_t^{(\ell)}\}_{t,\ell}$, we first run kmeans with M centers to find latent states $\mathcal{X} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_M\}$ that capture all key-steps and background subactions. Ideally, videos of the same task would share the same set of key-steps, i.e., there exists $k \ll M$ latent states that well represent videos of the same task. Notice that the set of k representative states for different tasks could have large or small/no overlap, depending on the similarity between the tasks. During training, the subset selection module takes the M states in $\mathcal{X}$ and the video features, $\{\boldsymbol{f}_t^{(\ell)}\}_{t,\ell}$, for all videos from the same task and selects k out of M centers that well represent the videos. To do so, we use a clustering-based subset selection method, where for the video of task $a \in \mathcal{A}$, we solve

$$\min_{\substack{\mathcal{S}\subseteq\{1,\ldots,M\}\\ |\mathcal{S}|\leq k}} J_a(\mathcal{S}) \triangleq \sum_{\ell:c_\ell=a} \frac{1}{T_\ell} \sum_{t=1}^{T_\ell} \min_{i\in\mathcal{S}} \|\boldsymbol{f}_t^{(\ell)} - \boldsymbol{x}_i\|_2, \tag{2}$$

whose outputs are the k selected states, corresponding to key-steps, and the assignment of each frame to each key-state. We denote by $r_t^\ell \in \{1, \ldots, M\}$, the index of the key-state (key-step) assigned to $\boldsymbol{f}_t^{(\ell)}$. This will be used for training the localization network, which we discuss in the next subsection.

To solve (2), we use a greedy algorithm, by considering an active set Γ that is incrementally grown to select at most k states. Initializing $\Gamma = \varnothing$, at each iteration, we add the state from $\mathcal{X}$ that minimizes the cost function the most compared to only using Γ. More specifically, we find $j^* \in \{1, \ldots, M\} \backslash \Gamma$ for which the gain, $\delta_\Gamma(j^*) \triangleq J_a(\Gamma) - J_a(\Gamma \cup j^*)$, is maximum. Thus, including the center j^* gives the largest decrease in the loss function. We include j^* in the active set and repeat the process k times.

Notice that subset selection allows different tasks to share some key-steps, e.g., it is expected that videos of 'making omelette' and 'making scrambled egg' share some key-steps, e.g., 'break egg', 'pour oil', etc.

2.1.3. Key-Step Localization Network (KLN). The KLN is a student deep neural network that learns to localize key-steps using pseudo labels provided by subset selection. More importantly, it allows backpropagating gradients for attention feature learning by bypassing the non-differentiable discrete subset selection module. More specifically, for a video ℓ with T_ℓ frames, KLN receives T_ℓ attention feature vectors, $\{\boldsymbol{f}_t^{(\ell)}\}_{t=1}^{T_\ell}$. The network then produces outputs $\{\boldsymbol{s}_t^{(\ell)}\}_{t=1}^{T_\ell}$ corresponding to T_ℓ frames, where each output $\boldsymbol{s}_t^{(\ell)} = \begin{bmatrix} s_{1,t}^{(\ell)} & s_{2,t}^{(\ell)} & \cdots & s_{M,t}^{(\ell)} \end{bmatrix}^\top$ is an M-dimensional vector specifying the score that the frame t of the video ℓ belongs to each of the M latent states in $\mathcal{X}$. We further normalize the scores using the

softmax function to obtain the probability of each segment t being assigned to each latent state $\boldsymbol{x}_m$ as

$$p_{m,t}^{(\ell)} \triangleq \frac{\exp\left(s_{m,t}^{(\ell)}\right)}{\sum_{n=1}^{M}\exp\left(s_{n,t}^{(\ell)}\right)},\ \forall m=1,\ldots,M. \tag{3}$$

In the experiments, we describe the exact architecture, consisting of a sequence of 1D convolutions, 1D pooling and 1D deconvolutions, used for the KLN. Given that training videos do not have key-step annotations and only contain video task labels, we use the output of the subset selection, which gives the M-dimensional one-hot encoding of each frame to each latent state (more precisely, the assignment of each frame to each of the k selected key-states) as pseudo labels to train the KLN; see the subsection for the loss function and the learning scheme.

Remark 2. While one can only use the task labels of videos as a form of weak supervision, we show in the experiments that key-step supervision from subset selection plays a pivotal role in improving the key-step localization performance.

2.1.4. Task Classification Network (TCN) with Learnable Key-Step Pooling. Our goal is to have a network that predicts the task label of a video, in addition to localization of key-steps. This also allows us to use the weak supervision, i.e., video label, for better localization of key-steps. To do so, we make two observations: i) *not all attention features should be used for video classification*: only the ones corresponding to key-steps should be used; ii) *not all attention features corresponding to key-steps are equally important* for video classification: the key-steps common across different tasks should be less emphasized, while discriminative key-steps more specific to the task should be more emphasized for video classification.

To take into account these observations, we first compute a key-step based attention feature for each of the M latent states by aggregating attention features that are assigned to each latent state by using the output of the KLN. More specifically, for each latent state m, we build a global *key-step based feature*

$$\boldsymbol{h}_m^{(\ell)} \triangleq \sum_{t=1}^{T_\ell} p_{m,t}^{(\ell)} \boldsymbol{f}_t^{(\ell)}, \tag{4}$$

by taking the weighted average of the frame attention features, where the weight of each frame is the probability that it belongs to the latent state m. Given M global feature, $\{\boldsymbol{h}_m^{(\ell)}\}_{m=1}^{M}$, we propose a learnable pooling mechanism that aggregates the features with different weights, by upweighting the discriminative ones for video classification and downweighting the non-discriminative ones. More specifically, we learn weights $\beta_m(\boldsymbol{h}_m^{(\ell)})$ for each key-step based attention feature and combine them to form the final video feature,

$$\bar{\boldsymbol{h}}^{(\ell)} \triangleq \sum_{m=1}^{M} \beta_m\big(\boldsymbol{h}_m^{(\ell)}\big)\boldsymbol{h}_m^{(\ell)}, \quad \beta_m\big(\boldsymbol{h}_m^{(\ell)}\big) = \frac{\exp(m_\beta(\boldsymbol{h}_m^{(\ell)}))}{\sum_{m'=1}^{M}\exp(m_\beta(\boldsymbol{h}_{m'}^{(\ell)}))}. \tag{5}$$

where the weights are parameterized by a neural network whose outputs $m_\beta(\boldsymbol{h}_m^{(\ell)})$ for given inputs $\{\boldsymbol{h}_m^{(\ell)}\}_{m=1}^M$ are normalized by a softmax function across all key-steps. This could also be seen as an *attention mechanism on the key-step based features*. Finally, we use the global video feature, $\bar{\boldsymbol{h}}^{(\ell)}$, as input to a fully connected network, whose output is the probability of the video belonging to each of the $|\mathcal{A}|$ tasks.

2.2 Proposed Learning Method

In order to train the parameters of the spatial attention network (SAN), key-step localization network (KLN) and Task Classification Network (TCN), using the key-step pseudo labels provided by the unsupervised subset selection module and using the ground-truth task labels, we propose to minimize a loss function that consists of the combination of a ranking loss for the key-step supervision and the cross-entropy loss for the video classification. More specifically, we minimize

$$\mathcal{L} \triangleq \mathcal{L}_{rank} + \lambda \mathcal{L}_{ce}, \quad \mathcal{L}_{ce} \triangleq -\sum_{\ell} \log p_\ell^{c_\ell}, \tag{6}$$

where $\mathcal{L}_{ce}$ is the cross-entropy loss that measures the consistency between the classification network's output and the ground-truth task label (p_ℓ^a is the probability of video ℓ belonging to task a). The ranking loss, $\mathcal{L}_{rank}$, promotes consistency between the outputs of the localization network and subset selection, i.e., for the t-th frame of the video ℓ, it must produce a higher score for the state $r_t^{(\ell)}$, which is the key-state obtained by subset selection to represent the segment t. Hence, we define

$$\mathcal{L}_{rank} \triangleq \sum_{\ell} \frac{1}{T_\ell} \sum_{t} \sum_{m \neq r_t^{(\ell)}} \max\big(0, 1 - s^{(\ell)}_{r_t^{(\ell)},t} + s^{(\ell)}_{m,t}\big), \tag{7}$$

whose minimization promotes to have the score of the pseudo ground-truth state $s^{(\ell)}_{r_t^{(\ell)},t}$ in the output of KLN be larger by a margin of one (other margins could be used as the network learns to automatically adjusts the weights accordingly) than the scores of other latent states.

We train our model by alternating between running subset selection and learning the DNN parameters. To find M states for subset selection, we use mini-batch kmeans [26], which finds cluster centers via gradient descent steps. Therefore, we refine the cluster centers while ensuring the consistency on the cluster assignments every time we rerun the kmeans (making sure the m-th center in an iteration of kmeans updates the m-th center in the previous iteration).

Remark 3. One could use a cross entropy loss instead of the ranking loss on the output of of KLN. In our experiments, however, the ranking loss performed better, as it is less restrictive, only enforcing the output of the KLN to give better score for the true key-step.

3 Experiments

3.1 Experimental Setup

Datasets. We perform experiments on ProceL [9] and CrossTask [37].The ProceL is a medium-scale dataset of 12 diverse tasks, such as *set up Chromecast*, *assemble clarinet* and *replace iPhone battery*. Each task consists of about 60 videos and on average contains 8 key-steps. CrossTask is a large instructional video dataset with 18 primary tasks ranging from cooking activities such as *make kimchi fried rice* to fixing car such as *jack up car*. There are 2,750 videos for these primary tasks with 7 key-steps per task on average. In both datasets, each task has a grammar of key-steps, e.g. 'perform CPR' consists of 'call emergency', 'check pulse', 'open airway', 'give compression' and 'give breath', where each video is annotated with the key-steps. Each video may not contain all the key-steps and the order of key-steps in some videos could be different, as there are multiple ways to perform the same task.

In the multi-task experiments, for ProceL, we randomly select videos in each task such that it has at least 10 videos for testing and at most 50 training videos. For CrossTask, we randomly select 70% of the videos in each task for training and the remaining 30% of videos are used for testing.

Evaluation Metrics. We measure the performance of key-step localization and task label classification as a function of the number of key-steps. For key-step localization, we report Recall and F1 score (Precision can be computed from Recall and F1). For Recall, we compute the ratio between the number of frames having correct key-step prediction and the number of ground-truth key-step frames across all key-steps. Precision is similar to recall except the denominator is the number of frames assigned to key-steps. F1 is the harmonic mean between Recall and Precision. Thus, unlike prior work [1, 9] that count 1 or 10 correct frame intersection as a true detection, we use the actual number of frames in the intersection to compute the scores, hence, the scores will be lower. For task classification, we use the standard classification accuracy over videos.

Implementation Details. We extract feature map from the last convolutional layer of VGG19 which has the size of $7 \times 7 \times 512$ as frame inputs to the model. We subsample 2 frames per second from each video to reduce the complexity of the model. In our framework, the spatial attention and the learnable key-step pooling each is parametrized by a neural network with 1 hidden layer of size equal to the input layer and the activation is set to hyperbolic tangent function. In the task classification network, once the global video feature is computed using the output of the learnable key-step pooling, we apply an additional layer to classify the task label of the given video.

To build the key-step localization network (KLN), we make connection between our goal, which is to take T_ℓ input vectors and output T_ℓ vectors each of dimension M, and semantic image segmentation whose goal is to take an input image and produce an output image where each pixel takes one of few discrete values corresponding to a category. Thus, we take the network in [17] and, given

that we are working with sequential data instead of 2D images, we convert 2D convolutions, 2D pooling and 2D deconvolutions to, respectively, 1D temporal convolutions, 1D pooling and 1D deconvolutions [24].

We implement our model in PyTorch and optimize with the default setting of RMSprop [30] with the learning rate 0.001, weight decay 0.0001 and batch size of 1 video. For the task-specific setting, where we use videos of each task separately to learn one model per task, we train all variants of our model for 10 and 5 epochs on ProceL and CrossTask, respectively, and use $M = 30$. In the multi-task setting, where we use all videos across all tasks to learn a single model, we train our framework with 3 and 2 epochs for ProceL and CrossTask, respectively. To optimize our model, we set $\lambda = 0.5$ for the multi-task setting ($\lambda = 0$ for single-task) and set $M = 50$ (we use larger M compared to the single-task setting as we have more key-steps collectively). Given that ProceL comes with segmentation of videos into superframes, we use segments and aggregate attention features in each segment as the input to DNNs and subset selection. For CrossTask, we use attention features of frames directly as inputs.

3.2 Experimental Results

We perform experiments for the *task-specific setting, learning a model for each task*, and the *multi-task setting, learning one model for videos across all tasks.*

3.2.1. Task-Specific Results. Given that existing unsupervised procedure learning algorithms work with videos of one task and learn a model separately for each task, we first perform experiments in this task-specific setting to investigate the effectiveness of our approach compared to the state of the art.[2] We compare with a simplified version of JointSeqFL [9] by ignoring the dynamical model (we refer to it as JointSeqFL-ND, where ND stands for no dynamics) and with the Temporal Embedding and Clustering (TEC) [16], as the two state-of-the-art methods that already have been shown to outperform other algorithms, including [1, 27]. We also compare with a simple baseline, called Random, where we predict the key-step labels of each video by randomly sampling prediction from a uniform distribution with k values, independently for each segment/frame. To have a fair comparison, we run our method on videos of each task *without the task classification network*, i.e., we do not use the task labels. Thus, KLN is only self-supervised by the subset selection module.

Tables 1 shows the average Recall and F1 scores (%) of different algorithms on the CrossTask dataset, as a function of the number of key-steps in each task. Notice that both our method and TEC perform better than JointSeqFL, which shows the importance of feature learning, as our method is self-supervised by subset selection, yet allows to learn attention features. Except $k = 7$, where the scores of our method are close to TEC, for other values of k, our method generally achieves much higher localization scores than TEC, especially for $k = 15$. Notice

[2] In this setting, there is no training and testing splits and all videos are used for learning and the localization performance is measured on all videos.

	$k=7$		$k=10$		$k=12$		$k=15$	
	Recall	F1	Recall	F1	Recall	F1	Recall	F1
Random	14.8	6.5	10.8	5.8	9.2	5.6	7.5	5.3
JseqFL-ND	29.6	12.5	26.6	12.6	23.0	12.1	21.3	12.4
TEC	**34.0**	**13.9**	28.5	13.0	27.2	**13.2**	25.5	12.7
Ours	33.3	13.4	**31.6**	**13.5**	**29.8**	13.0	**32.2**	**14.1**
Ours (multi-task)	*41.7*	*16.2*	*41.7*	*16.2*	*41.7*	*16.2*	*41.6*	*16.3*

Table 1: Recall and F1 (%) on CrossTask for different number of key-steps, k.

also that for all methods, the F1 score is much lower than Recall, which shows methods do better on recall than precision.

To investigate the effect of using videos from other tasks, we also show the results of running our method in the multi-task setting where the TCN is included and the classification loss is used in addition to the self-supervision (referred to as "*Ours (multi-task)*"). In this case, the localization accuracy of our method significantly improves while being less dependent on the value of k. This comes from the fact that taking advantage of other tasks allows to better discover the commonalities within the same task, by predicting the video label as well. Another advantage of this approach is that we will learn *one model across all tasks*, which we could later apply to any new video, while for the state of the art, one learns a separate model for each task and it is not clear how to localize key-steps of a new video for which the underlying task label is unknown.

3.2.2 Multi-Task Results. We consider the setting where we have videos of multiple tasks and our goal is to learn a single model that classifies the underlying task of a test video and recovers its segmentation according to assignments to key-steps.[3] We investigate the effect of different components of our framework as well as the effect of training using video label only versus training using both video labels and self-supervision.

Tables 2 and 3 show the Recall and F1 scores for key-step localization and task classification accuracy on ProceL and CrossTask, respectively. We show the effect of using spatial attention alone, learnable key-step pooling alone, and the combination of the two, where in all we use the supervision from both task class and subset selection. From the results, we make the following conclusions:

– The video classification accuracy on both datasets is always higher when using both spatial attention and learnable key-step pooling than using only one. For localization, the performance of different settings are close on ProceL, where there is no clear winner across all k's. We believe this is due to not having enough videos to effectively train the deep networks. On the other hand, on CrossTask that has a larger number of videos, the localization performance significantly improves when using both attention and learnable pooling, in particular, improving

[3] This is different and more challenging than weakly supervised learning from instructional videos [13, 23, 3, 5, 37], which assume knowing the list of key-steps in videos.

	$k=7$			$k=10$			$k=12$			$k=15$		
	R	F1	Acc	R	F1	Acc	R	F1	Acc	R	F1	Acc
Attention	22.0	11.3	88.3	**24.4**	**12.6**	89.6	**24.7**	**12.9**	88.3	20.9	10.4	89.2
Learn. Pool.	25.7	13.3	91.7	24.0	12.1	90.8	23.1	12.0	90.8	**24.5**	**12.5**	90.8
Both	**26.7**	**14.0**	**92.6**	23.8	12.4	**91.7**	23.8	12.8	**93.3**	22.8	11.8	**93.8**

Table 2: Localization scores (Recall, F1) and classification accuracy (Acc), in precent, of different algorithms on ProceL for different number of key-steps, k.

	$k=7$			$k=10$			$k=12$			$k=15$		
	R	F1	Acc	R	F1	Acc	R	F1	Acc	R	F1	Acc
Attention	39.2	16.0	74.3	27.6	11.1	71.7	28.3	11.4	71.7	34.9	13.8	73.8
Learn. Pool.	34.7	13.8	70.1	24.2	10.4	66.6	29.9	12.5	72.9	22.8	10.6	72.8
Both	**41.1**	**16.2**	**79.1**	**41.2**	**16.3**	**80.4**	**41.1**	**16.2**	**79.5**	**41.0**	**16.3**	**77.9**

Table 3: Localization scores (Recall, F1) and classification accuracy (Acc) of different algorithms on CrossTask for different number of key-steps, k.

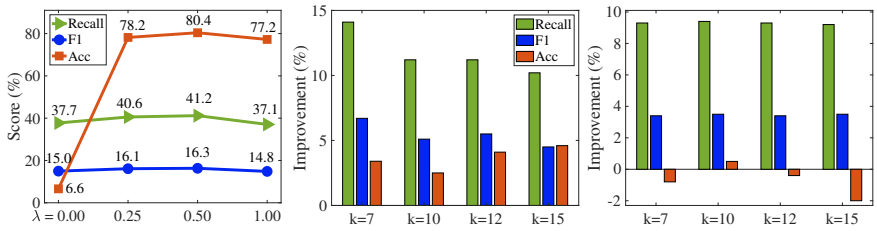

Fig. 2: Effect of the regularization parameter λ (left), effect of self-supervision for the multi-task setting on ProceL (middle) and CrossTask (right).

the F1 score by more than 5.7% for both $k \in \{10, 15\}$ and more than 2.4% for $k \in \{7, 12\}$. Notice that, similar to the task-specific setting, the performance of our method on CrossTask is robust for different values of k.

– Given that CrossTask has a larger number of videos per task than ProceL, the localization is generally higher on CrossTask than ProceL, as the DNN benefits from having more training videos. On the other hand, the task classification accuracy is higher on ProceL than CrossTask. This comes from the fact that the 12 tasks in ProceL are diverse with less overlap (except two about cooking and two about fixing cars), while the18 tasks in CrossTask have more overlap (e.g., several on cooking and several on making drinks).

Effect of Self-Supervision and Hyperparameter. Figure 2 (left) shows the effect of the regularization parameter λ in (6) on the localization performance (Recall and F1) and classification accuracy (Acc) on CrossTask for $k = 10$. Notice that while all scores are maximum for $\lambda = 0.5$, achieving Recall= 41.2, F1= 16.3 and Acc=80.4, the performance does not change much for other values of λ, e.g., achieving Recall= 40.6, F1= 16.1 and Acc=78.2 for $\lambda = 0.25$. As

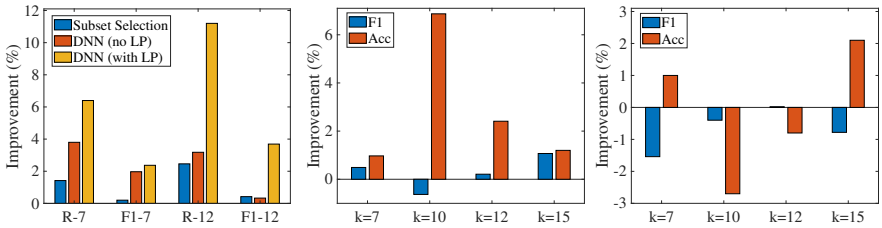

Fig. 3: Improvement obtained by learning attention features on CrossTask (left), improvement obtained by fine tuning when learning attention features on ProceL (middle) and CrossTask (right). All results are for the multi-task setting.

expected, when $\lambda = 0$ (no task classification loss), Acc is low. Figure 2 (middle, right) show the improvement obtained by using both self-supervision via subset selection and task labels of videos over only using task labels on ProceL (middle) and CrossTask (right). Notice that the localization performance significantly improves on both datasets when using the self supervision. The classification accuracy always improves on ProceL with smaller number of videos, as subset selection provides more supervision to train the deep architecture, including the classification network. On the other hand, the classification accuracy on CrossTask slightly decreases. This comes from the fact that there are already enough videos for effective training of the video classifier, hence, including the localization loss puts less emphasis on the video classification performance.

3.2.3. Effect of Attention Feature Learning and Fine Tuning. We used attention features to automatically focus on informative regions (where *human and object(s) interact*) to extract feature from. These features are not only given to the DNN, but also to the subset selection that provides supervision for the localization. Figure 3 (left) shows the Recall and F1 improvement obtained by learning attention features over not using any attention for $k = 7$ and $k = 12$ for the multi-task setting on CrossTask. We show the improvement for subset selection as well as the deep network without and with learnable pooling (LP). In all cases, using attention improves the performance. In other words, using the subset selection supervision and back-propagating gradient via the differentiable network, we obtain better features that subsequently improve the performance of not only DNN but the subset selection itself. It is also worth noting that in all cases, DNN enjoys higher improvement than subset selection, thanks to the capacity and generalization power of the DNN compared to subset selection that works directly on centroids. Figure 3 (middle, right) show the effect of fine-tuning the last two layers of the VGG network on ProceL (middle) and CrossTask (right) for the multi-task setting. Notice that fine-tuning on ProceL generally improves the performance, while on CrossTask it could improve or degrade the performance. In all cases, the localization performance slightly changes, which shows attention feature learning already provides sufficient information for localization.

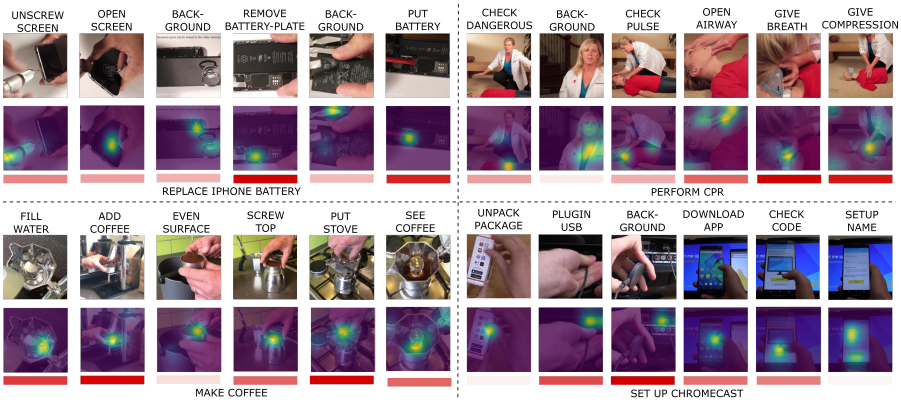

Fig. 4: Visualization of the spatial attention and strength of key-step weights (shown below spatial attention) for videos from four tasks. Notice that, in general, our method successfully focuses on the informative regions of frames and the learnable pooling gives higher weights to more informative key-steps.

3.2.4. Qualitative Results. Figure 4 shows the learned spatial attention of our network and the strength of key-steps (the product between the probability of the most probable latent state in a frame and the key-step weight corresponding to that latent state) for videos from four tasks in ProceL. Notice that in all, our method attends to informative regions, e.g., in 'change iPhone battery', it attends to 'screen' or 'battery' in the associated key-steps. Similarly, for 'CPR', our attention learns to focus on the patient's neck for 'check pulse' or his mouth/chest for 'give breath'/'give compression'. On the other hand, to recognize the task, the learnable key-step pooling gives higher weights to actual key-steps and less weights to background frames. Most notably, it gives the largest weights to 'give breath' and 'give compression' in CPR, which are the most discriminative steps of CPR. In 'setup Chromecast' where some of the steps are visually very similar, however, it gives a high weight to the background, reducing the performance.

4 Conclusions

We developed a self-supervised multi-task procedure learning method that allows to learn a single deep neural network (DNN) for discovering key-steps and task classification using training videos from multiple tasks. By experiments on two instructional video datasets, we showed the effectiveness of our method for unsupervised discovery of procedure steps and video classification.

Acknowledgements

This work is partially supported by DARPA Young Faculty Award (D18AP00050), NSF (IIS-1657197), ONR (N000141812132) and ARO (W911NF1810300).

References

1. Alayrac, J.B., Bojanowski, P., Agrawal, N., Sivic, J., Laptev, I., Lacoste-Julien, S.: Unsupervised learning from narrated instruction videos. IEEE Conference on Computer Vision and Pattern Recognition (2016)
2. Ba, J., Caruana, R.: Do deep nets really need to be deep? Neural Information Processing Systems (2013)
3. Bojanowski, P., Lajugie, R., Bach, F., Laptev, I., Ponce, J., Schmid, C., Sivic, J.: Weakly supervised action labeling in videos under ordering constraints. European Conference on Computer Vision (2014)
4. Caron, M., Bojanowski, P., Joulin, A., Douze, M.: Deep clustering for unsupervised learning of visual features. European Conference on Computer Vision (2018)
5. Ding, L., Xu, C.: Weakly-supervised action segmentation with iterative soft boundary assignment. IEEE Conference on Computer Vision and Pattern Recognition (2018)
6. Du, X., Mishra, B.D., Tandon, N., Bosselut, A., tau Yih, W., Clark, P., Cardie, C.: Weakly-supervised action segmentation with iterative soft boundary assignment. Annual Meeting of the North American Association for Computational Linguistics (2019)
7. Elhamifar, E.: Sequential facility location: Approximate submodularity and greedy algorithm. International Conference on Machine Learning (2019)
8. Elhamifar, E., De-Paolis-Kaluza, M.C.: Subset selection and summarization in sequential data. Neural Information Processing Systems (2017)
9. Elhamifar, E., Naing, Z.: Unsupervised procedure learning via joint dynamic summarization. International Conference on Computer Vision (2019)
10. Goel, K., Brunskill, E.: Learning procedural abstractions and evaluating discrete latent temporal structure. International Conference on Learning Representation (2019)
11. Hinton, G.E., Vinyals, O., Dean, J.: Distilling the knowledge in a neural network. ArXiv (2015)
12. Huang, D., Buch, S., Dery, L., Garg, A., Fei-Fei, L., Niebles, J.C.: Finding it?: Weakly-supervised reference-aware visual grounding in instructional videos. IEEE Conference on Computer Vision and Pattern Recognition (2018)
13. Huang, D.A., Fei-Fei, L., Niebles, J.C.: Connectionist temporal modeling for weakly supervised action labeling. European Conference on Computer Vision (2016)
14. Huynh, D., Elhamifar, E.: Fine-grained generalized zero-shot learning via dense attribute-based attention. IEEE Conference on Computer Vision and Pattern Recognition (2020)
15. Huynh, D., Elhamifar, E.: A shared multi-attention framework for multi-label zero-shot learning. IEEE Conference on Computer Vision and Pattern Recognition (2020)
16. Kukleva, A., Kuehne, H., Sener, F., Gall, J.: Unsupervised learning of action classes with continuous temporal embedding. IEEE Conference on Computer Vision and Pattern Recognition (2019)
17. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. IEEE Conference on Computer Vision and Pattern Recognition (2015)
18. Malmaud, J., Huang, J., Rathod, V., Johnston, N., Rabinovich, A., Murphy, K.: What's cookin'? interpreting cooking videos using text, speech and vision. NAACL (2015)

19. Miech, A., Alayrac, J.B., Smaira, L., Laptev, I., Sivic, J., Zisserman, A.: End-to-end learning of visual representations from uncurated instructional videos. arXiv:1912.06430 (2019)
20. Miech, A., Zhukov, D., Alayrac, J.B., Tapaswi, M., Laptev, I., Sivic, J.: Howto100m: Learning a text-video embedding by watching hundred million narrated video clips. International Conference on Computer Vision (2019)
21. Phuong, M., Lampert, C.: Towards understanding knowledge distillation. International Conference on Machine learning (2019)
22. Puig, X., Ra, K., Boben, M., Li, J., , Wang, T., Fidler, S., Torralba, A.: Virtualhome: Simulating household activities via programs. IEEE Conference on computer Vision and Pattern Recognition (2018)
23. Richard, A., Kuehne, H., Gall, J.: Weakly supervised action learning with rnn based fine-to-coarse modeling. IEEE Conference on Computer Vision and Pattern Recognition (2017)
24. Rochan, M., Ye, L., Wang, Y.: Video summarization using fully convolutional sequence networks. European Conference on Computer Vision (2018)
25. Romero, A., Ballas, N., Kahou, S.E., Chassang, A., Gatta, C., Bengio, Y.: Fitnets: Hints for thin deep nets. International Conference on Learning Representations (2014)
26. Sculley, D.: Web-scale k-means clustering. WWW (2010)
27. Sener, F., Yao, A.: Unsupervised learning and segmentation of complex activities from video. IEEE Conference on Computer Vision and Pattern Recognition (2018)
28. Sener, F., Yao, A.: Zero-shot anticipation for instructional activities. International Conference on Computer Vision (2019)
29. Sener, O., Zamir, A.R., Savarese, S., Saxena, A.: Unsupervised semantic parsing of video collections. IEEE International Conference on Computer Vision (2015)
30. Tijmen, T., Hinton, G.: Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. COURSERA: Neural networks for machine learning 4.2 (2012)
31. Xu, C., Elhamifar, E.: Deep supervised summarization: Algorithm and application to learning instructions. Neural Information Processing Systems (2019)
32. Xu, K., Ba, J., Kiros, R., Cho, K., Courville, A.C., Salakhutdinov, R.R., Zemel, R.S., Bengio, Y.: Show, attend and tell: Neural image caption generation with visual attention (2015)
33. Yang, Y., Li, Y., Fermüller, C., Aloimonos, Y.: Robot learning manipulation action plans by watching unconstrained videos from the world wide web. AAAI (2015)
34. Yu, S.I., Jiang, L., Hauptmann, A.: Instructional videos for unsupervised harvesting and learning of action examples. ACM International Conference on Multimedia (2014)
35. Zagoruyko, S., Komodakis, N.: Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer. International Conference on Learning Representations (2017)
36. Zhou, L., Xu, C., Corso, J.J.: Towards automatic learning of procedures from web instructional videos. AAAI (2018)
37. Zhukov, D., Alayrac, J.B., Cinbis, R.G., Fouhey, D., Laptev, I., Sivic, J.: Cross-task weakly supervised learning from instructional videos. IEEE Conference on Computer Vision and Pattern Recognition (2019)