I^3: Interactive Iterative Improvement for Few-Shot Action Segmentation

Martina Gassen[1,*], Frederic Metzler[1,*], Erik Prescher[1,*],
Lisa Scherf[1,2], Vignesh Prasad[3,4], Felix Kaiser[3], Dorothea Koert[1,2]

***Abstract*— Extracting modular segments from raw video demonstrations of high-level actions is important to understand the underlying building blocks for different tasks in human-robot interaction. While (data-hungry) supervised learning approaches for Action Segmentation show good performance when the underlying segments are predefined, their performance degrades when unseen actions are introduced on-the-go as new data samples are scarce. In this regard, Zero- and Few-Shot Learning approaches have shown good performance in generalizing to unseen examples. In Action Segmentation, where each frame needs to be labeled, annotating new data even for a few tasks can become tedious as the number of tasks scale. In this work, we propose Interactive Iterative Improvement (I^3) for Few-Shot Action Segmentation, a Semi-Supervised Interactive Meta-Learning approach for Zero-Shot Learning on unlabeled videos and Few-Shot Learning on small amounts of labeled videos. I^3 consists of a Prototypical Network model for frame-wise prediction coupled with a Hidden-Semi-Markov-Model to prevent over-segmentation. The model is iteratively improved in an interactive manner through users' annotations provided via a webinterface. This is done in a task-agnostic manner that, in theory, can be reused for a number of different actions. Our model provides sequentially accurate segmentations using only a limited amount of labeled data which shows the efficacy of our learning approach. A lower edit distance compared to baselines indicates a lower number of required user edits making it well suited for non-expert users to smoothly provide annotations enabling them to have more control over the learned model.**

***Index Terms*— Interactive Learning, Action Segmentation, Few-Shot Learning, Human-In-The-Loop**

I. Introduction

The ability to recognize human actions is crucial for future assistive robotics [34] and an essential building block for robot behavior in more complex pipelines [9]. Similar to other fields of computer vision, like pose estimation [4] and image segmentation [19], the advancements of Deep Learning approaches have also been successfully shown in action classification on segmented videos [6, 13]. While in traditional scenarios the action recognition problem may be well defined with pre-known tasks and readily available datasets, there is a need for an ad hoc action recognition solution that can be easily adapted to custom datasets.

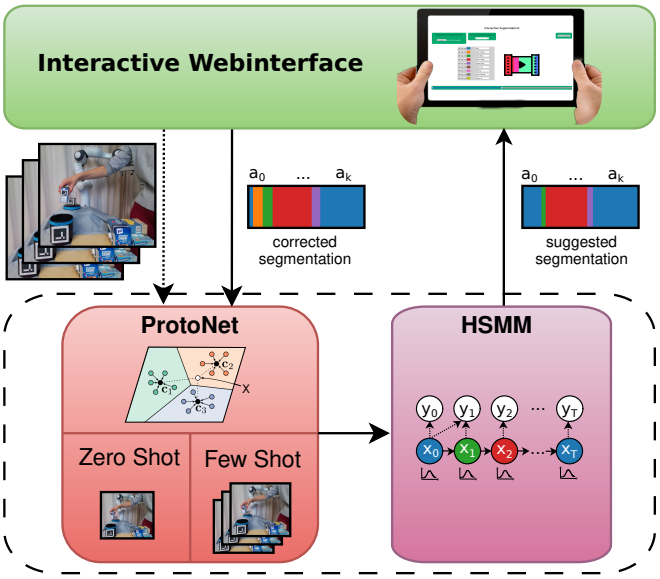


Fig. 1: We propose a framework for Iterative Interactive Improvement for Few-Shot Action Segmentation. When an action sequence is first provided, Zero-Shot Learning is used (dotted line) to provide an initial segmentation which is then iteratively improved using Few-Shot Learning (solid line). During an improvement iteration, the user corrects a predicted segmentation which is then used to train the model.

Although supervised methods have been showing excellent performance in the first conventional scenario [2, 32, 37], they require huge amounts of data and frame-by-frame labels, which is unfeasible for interactive settings. In contrast, unsupervised methods [31, 25] are well suited for the task but generally underperform their supervised counterparts.

In this regard, Few-Shot Learning has shown good performance with good data efficiency such that a model is able to generalize to unseen samples/classes given only a few or even a single (in the case of One-Shot Learning) example. Such a paradigm coupled with interactive improvement has been commonly used for image segmentation [3, 14, 29], gesture recognition [42] text synthesis (see [8] for an overview) and spoken language understanding [15] where user inputs in the form of annotations, prompts or labels enable the model to adjust and correct themselves.

We introduce a novel framework I^3: Interactive Iterative Improvement for Few-Shot Action Segmentation following the aforementioned idea of interactive Few-Shot Learning. The framework aims to bridge the gap between un-

[1] Interactive AI & Cognitive Models for Human-AI Interaction (IKIDA), TU Darmstadt, Germany; [2] Centre for Cognitive Science, TU Darmstadt, Germany; [3] Institute for Intelligent Autonomous Systems, Department of Computer Science, TU Darmstadt, Germany; [4] Chair for Marketing and Human Resource Management, Department of Law and Economics, TU Darmstadt, Germany
[*] - Equal contribution
This work was funded by the German Federal Ministry of Education and Research (BMBF) projects IKIDA (Grant No. 01IS20045) and KompAKI (Grant No. 02L19C150) and the European Union's Horizon 2021 EIC Transition programme project "Visual Robot Programming" (Grant No. 101058252). Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or EIC Transition. Neither the European Union nor the granting authority can be held responsible for them.

supervised and supervised solutions for Action Segmentation in a Zero-/Few-Shot manner using interactive improvement of the model from an end user.

Our approach consists primarily of two components: A Prototypical Network (ProtoNet) that generates a segmentation of the input video and a Hidden-Semi-Markov-Model (HSMM) that smooths the segmentation and reduces frequent, undesirable changes of predicted action segments. The ProtoNet applies Zero-Shot Learning to an unlabeled and unseen video. The output predictions of the Zero-Shot approach are then interactively adjusted by the user. Subsequently, these adjustments are reused for Few-Shot Learning to predict additional videos.

Overall, the main contribution of this paper is an interactive learning framework for temporal Action Segmentation. We focus specifically on the limited availability of labeled data obtained during run-time via user-annotated sequences. Here, we leverage Zero-/Few-Shot Learning via a ProtoNet in combination with an HSMM to smooth out the predicted segmentation. In addition to the algorithmic back-end, we also develop a user interface to interactively improve the pre-trained Action Segmentation model in a Few-Shot manner.

II. Related Work

Action Segmentation: Action Segmentation refers to the classification of a temporal sequence of video/skeleton/sensor readings frame-by-frame. Various methods have been proposed, ranging from Recurrent Neural Networks and Hidden-Markov-Models (HMM) [24] to coarse-to-fine encoder-decoder ensembles [32], transformer-based approaches [2] and Gaussian Processes [27]. Such methods are prone to over-segmentation, which can typically be mitigated by smoothing an intermediate result using regularization. In unsupervised settings, clustering the features of each frame or different temporal embeddings over frames [25, 31] is common. Enforcing the number of occurrences of the action labels to prevent over-segmentation may also be an option. However, the feasibility of such methods depends heavily on the dataset.

For supervised approaches one could use labeling from expert annotators [17, 23, 11], but this can be time consuming and expensive. While crowd-sourcing is an alternative, it involves non-expert annotators. The data must, therefore, be validated manually which this work intends to avoid. We circumvent this by having users interact with the system directly, giving them direct control over the data quality.

Another common post-processing technique for tackling over-segmentation is Viterbi decoding [30, 18, 24, 2], a dynamic programming approach that returns the most likely sequence of hidden states by modelling a sequence with an HMM to smooth out an over-segmented sequence. Alternatively, over-segmentation can be reduced by either learning the length of each segment through an additional model [30, 2] or penalizing incorrect segment lengths [26, 2]. Semi-supervised methods [18, 30, 26, 41] mainly focus on additional losses and constraints to reduce the required labeled data for each frame. They appear to greatly improve overall performance. Therefore, we explore such a semi-supervised paradigm in a more aggressive manner with regards to data scarcity by leveraging Few-Shot Learning.

Interactive Few-Shot Learning: Few-Shot Learning [16, 12] is a machine learning paradigm where a model learns from limited data, typically in a supervised manner, for a given task and has become a popular paradigm in computer vision, natural language processing, robot learning, and more [39]. In cases where annotated data is not always readily available, one way to circumvent this issue is to enable an interactive Human-in-the-Loop approach of providing labels for the Few-Shot case. This has been extensively explored in medical image segmentation not just for providing labels but also in correcting wrong predictions that the model parameters are trained with [35]. The aforementioned techniques can - among others - also be found in other vision [22, 36] and robot learning [38, 1, 20] tasks.

The key takeaway from this line of work is that interactive methods capture the benefits of fully automated learning-based approaches while giving the user sufficient control over the outputs [35], thereby making interactive Few-Shot Learning a suited option to explore for iteratively improving Action Segmentation. A general overview on interactive Human-in-the-loop Machine Learning can be found in [40] and on Few-Shot Learning can be found in [39].

III. I^3: Interactive Iterative Improvement for Few-Shot Action Segmentation

In this section, we introduce our method I^3: Interactive Iterative Improvement for Few-Shot Action Segmentation. It consists of an interface through which videos can be uploaded and segmented, as outlined in Section III-C. We suggest an initial segmentation to the user, which improves with each segmented video. To gain data from videos of variable length, we use an automated way of feature creation, as well as handcrafted features. The features $[\boldsymbol{x}_t]_{t=1\dots T}, \boldsymbol{x}_t \in \mathbb{R}^{D_x}$ of a frame at time t get classified as an action segment $k \in \{1 \dots K\}$. Initially, these features are used to segment the first video with unsupervised Zero-Shot Learning. This newly predicted segmentation is corrected by the user and the resulting newly labeled data is used to refine the model. After the user provides a corrected segmentation of the first video, each following video is iteratively segmented using Few-Shot Learning. The model consists of a ProtoNet (Section III-A) and an HSMM to smooth and correct the ProtoNets prediction (Section III-B). The method is intended to be data and run-time efficient, resulting in a tool that allows the user to segment a series of videos more easily and accurately. A flowchart of this segmentation process is visualized in Figure 1.

A. Prototypical Network for Few-Shot Learning

The backbone classification is performed by a Prototypical Network (ProtoNet) [33], which has been adapted for sequential data by using a recurrent structure and a temporal window of inputs. The main task of the network $f_{\boldsymbol{\theta}}(\cdot)$ is to

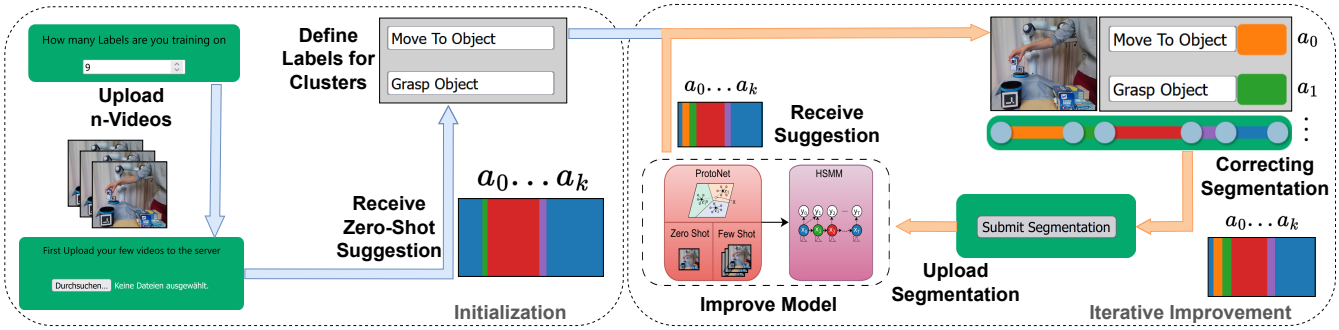


Fig. 2: A flowchart representing the iterative interaction by the user with the webinterface. The initial steps are indicated with blue arrows, while the steps of the iterative improvement loop are highlighted with orange.

encode the inputs with an encoding function into a D_{enc}-dimensional embedding space. At time t, given a sequence of of features $\boldsymbol{x}_{t-w:t}$ of a temporal window of size w, they are first encoded by the network $\boldsymbol{z}_t = f_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-w:t})$ into the embedding space. Using the embeddings of all samples, a set of prototypes $\boldsymbol{C}_{\boldsymbol{\theta}} = \{\boldsymbol{c}_k | k = 1 \dots K\}$ are computed which denote a representation of the classes in the embedding space. Computing the prototypes can be different for different tasks. Using the embeddings $\boldsymbol{z}_t$ and the set of prototypes $\boldsymbol{C}_{\boldsymbol{\theta}}$, the probability of the predicted label y_t being assigned as the class k can be computed by the softmax over the negative distances from the encoding to the prototypes:

$$P_{\boldsymbol{\theta}}(y_t = k | \boldsymbol{x}_{t-w:t}; \boldsymbol{C}_{\boldsymbol{\theta}}) = \frac{\exp(-d(\boldsymbol{z}_t, \boldsymbol{c}_k))}{\sum_{i=1}^{K} \exp(-d(\boldsymbol{z}_t, \boldsymbol{c}_i))} \quad (1)$$

where $d : \mathbb{R}^{D_{enc}} \rightarrow \mathbb{R}^+ \cup \{0\}$ can be any suitable distance metric e.g. the euclidean distance. We denote this for an entire set of inputs $\boldsymbol{X} = \{\boldsymbol{x}_{t-w:t}\}_{t=w \dots T}$ as $\hat{\boldsymbol{Y}} = \boldsymbol{P}_{\boldsymbol{\theta}}(\boldsymbol{X})$.

For Few-Shot Learning as well as for pre-training, each batch is randomly split into a support-set $\boldsymbol{S}$ and a query-set $\boldsymbol{Q}$. The prototypes are then computed by taking the class-wise means of the encodings of samples from the support-set

$$\boldsymbol{c}_k = \frac{1}{|\boldsymbol{S}_k|} \sum_{\boldsymbol{x}_s \in \boldsymbol{S}_k} f_{\boldsymbol{\theta}}(\boldsymbol{x}_s) \quad (2)$$

where $\boldsymbol{S}_k \subset \boldsymbol{S}$ contains only inputs labeled with class k. The network is trained by minimizing the Negative-Log-Likelihood of Eq. 1 with samples from the query-set $\boldsymbol{x}_q \in \boldsymbol{Q}$ classified against prototypes $\boldsymbol{c}_k$ from the support-set (Eq. 2).

$$\mathcal{L}_{\boldsymbol{\theta}}(\boldsymbol{S}, \boldsymbol{Q}) = \frac{1}{|\boldsymbol{Q}|} \sum_{\boldsymbol{x}_q \in \boldsymbol{Q}} -\log P(y = k | \boldsymbol{x}_q; \boldsymbol{C}_{\boldsymbol{\theta}}(\boldsymbol{S})) \quad (3)$$

$$\boldsymbol{\theta} = \arg\min_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{S}, \boldsymbol{Q}) \quad \forall \boldsymbol{S}, \boldsymbol{Q} \in \boldsymbol{D} \quad (4)$$

where $\boldsymbol{D}$ is the labeled dataset and $\boldsymbol{C}_{\boldsymbol{\theta}}(\boldsymbol{S})$ is the set of prototypes for a given support set $\boldsymbol{S}$ according to Eq. 2.

For Zero-Shot Learning there is no data which can be divided into a query- and a support-set, thus a different approach is needed. The original work suggests producing the prototypes by embedding meta-data which can, for example, come from a textual description of the class. However, this requires a separate trained model, which in turn increases the amount of training data needed. As we aim to be data-efficient, this is not viable in our case. Instead, we perform k-means clustering on the encoded data and use the resulting cluster centers as the class prototypes. Notably, the resulting labels are relative to the cluster centers and thus need to be mapped to the actual labels, which we achieve by the human agent in the loop.

B. Hidden (Semi-) Markov Model

We employ a Hidden Semi-Markov Model [43] to smooth and improve the ProtoNet's predictions. An HMM is denoted by a set of hidden states $h \in \{1, \dots, H\}$ with an intial state distribution π_h and state transition probabilities $\mathcal{T}_{i,j}$ of changing from state i to j and a set of observations $\boldsymbol{Y} = \{\boldsymbol{y} \in \mathbb{R}^{D_y}\}$ that can be emitted at each time t from a state h with a given emission probability. In our use case, the states correspond to the set of possible actions $\{1, \dots, K\}$, and the observations correspond to the ProtoNet's prediction $\hat{\boldsymbol{y}}_t = [P(y = h | \boldsymbol{x}_{t-w:t}; \boldsymbol{C}_{\boldsymbol{\theta}})]_{h=1 \dots H}$ (Eq. 1). We characterize the emission probabilities under state h via a Gaussian distribution $\mathcal{N}(\hat{\boldsymbol{y}}_t; \boldsymbol{\mu}_h, \boldsymbol{\Sigma}_h)$ with the class-wise mean $\boldsymbol{\mu}_h$ and the covariance $\boldsymbol{\Sigma}_h$ of the observations $\hat{\boldsymbol{y}}_t$. An HMM is, therefore, defined with the parameters $\boldsymbol{\psi} = \{\pi_h, \mathcal{T}_{i,j}, \boldsymbol{\mu_h}, \boldsymbol{\Sigma_h}\}$ This allows us to predict the segmentation using the forward variable of the HMM:

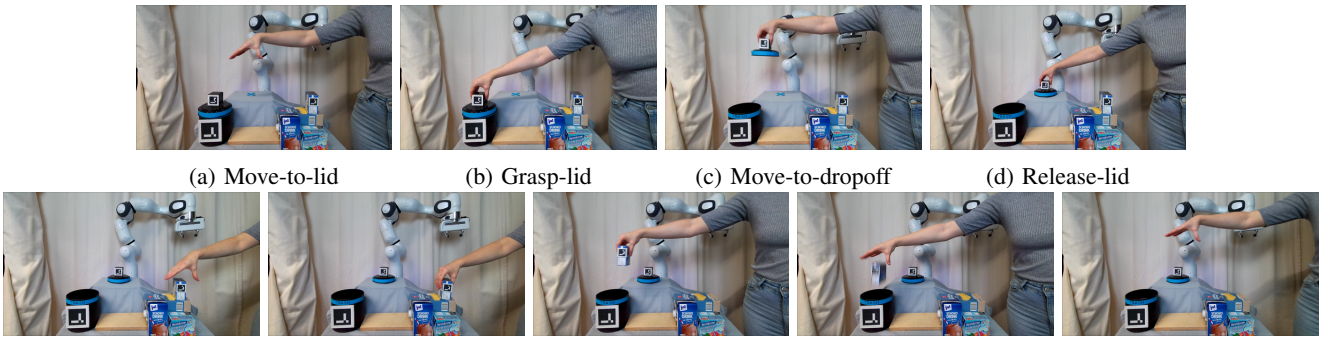
$$\arg\max_h \frac{\alpha_h(\hat{\boldsymbol{y}}_t)}{\sum_{i=1}^{H} \alpha_i(\hat{\boldsymbol{y}}_t)} \quad (5)$$

$$\alpha_h(\hat{\boldsymbol{y}}_t) = \mathcal{N}(\hat{\boldsymbol{y}}_t; \boldsymbol{\mu}_h, \boldsymbol{\Sigma}_h) \sum_{i=1}^{H} \alpha_i(\hat{\boldsymbol{y}}_{t-1}) \cdot \mathcal{T}_{i,h} \quad (6)$$

where $\alpha_h(\hat{\boldsymbol{y}}_0) = \pi_h$. Using this, we denote the segmentation obtained for an entire sequence as $\hat{\boldsymbol{Y}}^* = \boldsymbol{H}_{\boldsymbol{\psi}}(\boldsymbol{P}_{\boldsymbol{\theta}}(\boldsymbol{X}))$

An HSMM is a special form of a Hidden Markov Model, with a relaxation of the Markov rule where current state depends not only on the previous state and the current observation but also on duration $d \in 1 \dots \tau$ spent in the current state. This is characterised by an additional distributions $p_h(d)$ over the durations d for each state h:

$$\alpha_h(\hat{\boldsymbol{y}}_t) = \mathcal{N}(\hat{\boldsymbol{y}}_t; \boldsymbol{\mu}_h, \boldsymbol{\Sigma}_h) \sum_{i=1}^{H} \sum_{d=1}^{\tau} \alpha_i(\hat{\boldsymbol{y}}_{t-1}) \cdot \mathcal{T}_{i,h} \cdot p_h(d) \quad (7)$$



(a) Move-to-lid (b) Grasp-lid (c) Move-to-dropoff (d) Release-lid

(e) Move-to-trash (f) Grasp-trash (g) Move-to-trashcan (h) Release-trash (i) Retrieving

Fig. 3: Images of each of the nine different segments in a video. Segments (a)-(d) are exclusive to the *lid* complexity type, while (e)-(i) can be found in both *lid* and *simple*.

This time-dependent definition also allows the modeling of random dependencies w.r.t. the length of the action segments. The HSMM parameters are trained using Expectation Maximisation (EM), initially using the predictions from the Zero-Shot segmentation as ground truth followed by the user annotated labels in the subsequent iterations. An in-depth explanation of HMMs, HSMMs and their training can be found in [5, 28], which we omit due to space constraints.

C. User Interface for Iterative Refinement

In order to iteratively improve the underlying model with human input, a webinterface is implemented that allows users to upload a variable number of videos and optimize the segmentation of each video. For the first video, the interface initially suggests a Zero-Shot segmentation. The one-hot versions of these predicted labels are used to initialize the HSMM. After smoothing the predictions, they are presented in the webinterface. The user can then make any necessary adjustments to the segmentation. This annotated segmentation is then added to the dataset and is used to improve the suggestion for the next video by training both the ProtoNet and the HSMM with the updated dataset. This process is repeated until all uploaded videos are segmented. Since the suggestions for each video are more precise, the amount of time and effort required by the user is reduced at each iteration. An overview of the interactive framework is shown in Figure 2, as well in Algorithm 1.

Algorithm 1: I^3 : Interactive Iterative Improvement

Input: Pre-trained ProtoNet $P_{\boldsymbol{\theta}}$, Features from N videos $\{\boldsymbol{X}_n = \{\boldsymbol{x}_{t-w:t}\}_{t=i\dots T_n}\}_{n=1\dots N}$
Output: ProtoNet $P_{\boldsymbol{\theta}}$, HSMM $H_{\boldsymbol{\psi}}$
$\hat{\boldsymbol{Y}}_1^{ZS} = \boldsymbol{P}_{\boldsymbol{\theta}}(\boldsymbol{X}_1)$; `# Initially Zero-Shot`
$\hat{\boldsymbol{Y}}_1^{OH} = \text{one-hot}(\hat{\boldsymbol{Y}}_1^{ZS})$
Update $\boldsymbol{\psi}$ using EM on $(\hat{\boldsymbol{Y}}_1^{ZS}, \hat{\boldsymbol{Y}}_1^{OH})$ as in [5, 28]
$\hat{\boldsymbol{Y}}_1 = \boldsymbol{H}_{\boldsymbol{\psi}}(\hat{\boldsymbol{Y}}_1^{ZS})$
$\boldsymbol{Y}_1^* \leftarrow$ User Labels from $H_{\boldsymbol{\psi}}(\hat{\boldsymbol{Y}}_1)$
$\boldsymbol{D} \leftarrow \{(\boldsymbol{X}_1, \boldsymbol{Y}_1^*)\}$; `# Labeled Data`
for $n = 2 \dots N$ **do**
  $\hat{\boldsymbol{Y}}_n \leftarrow \boldsymbol{H}_{\boldsymbol{\psi}}(\boldsymbol{P}_{\boldsymbol{\theta}}(\boldsymbol{X}_n))$; `# Few-Shot`
  $\boldsymbol{Y}_n^* \leftarrow$ User Labels from $\hat{\boldsymbol{Y}}_n$
  $\boldsymbol{D} \leftarrow \boldsymbol{D} \cup \{(\boldsymbol{X}_n, \boldsymbol{Y}_n^*)\}$
  Split $\boldsymbol{D}$ into $\boldsymbol{S}$ and $\boldsymbol{Q}$
  Update $\boldsymbol{\theta}$ using Eq. 3 and 4 using $\boldsymbol{S}, \boldsymbol{Q}$
  $\boldsymbol{D_H} = \bigcup_{\boldsymbol{X}_j, \boldsymbol{Y}_j \in \boldsymbol{D}} (\boldsymbol{P}_{\boldsymbol{\theta}}(\boldsymbol{X}_j), \boldsymbol{Y}_j)$
  Update $\boldsymbol{\psi}$ using EM on $\boldsymbol{D_H}$ as in [5, 28]
end
return: $\boldsymbol{P}_{\boldsymbol{\theta}}$, $H_{\boldsymbol{\psi}}$

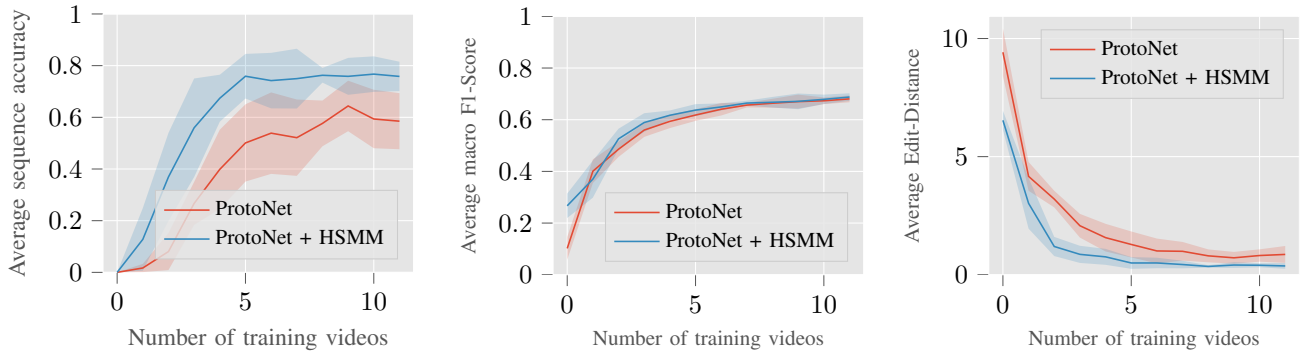
IV. Experiments

Our model is evaluated on a dataset of humans demonstrating a trash disposal task with two different levels of complexity, as described in Section IV-A. The details of our implementation and the baselines used are outlined in Section IV-B following which we discuss our results in Section IV-C.

A. Dataset

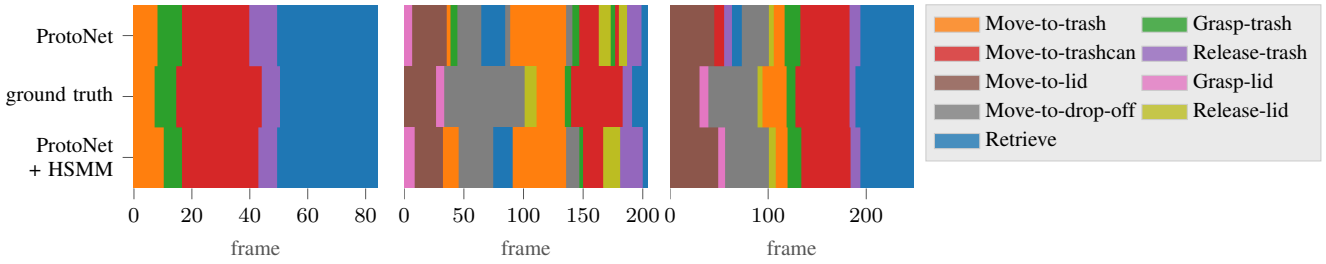
The dataset used for evaluation consists of RGB-D videos of 23 people performing a trash disposal task. Each person provided three demonstrations for two levels of complexity: *simple* and *lid*. In the *simple* setting, the participants demonstrated how to pick up some trash from varying starting positions and put it in a trashcan. In the more complex *lid* setting, a lid was placed on the trashcan, which had to be removed before the trash could be disposed of. The experiments were approved by the ethic committee of TU-Darmstadt on 07/15/2022.

1) Segments: The videos were manually labeled on a framewise level with the corresponding high-level action. These labels serve as the ground truth segments in the following experiments. All nine segments included in the dataset are visualized in Figure 3. In the *simple* setting, the task demonstrations can be segmented into five actions: Move-to-trash, Grasp-trash, Move-to-trashcan, Release-trash, and Retrieving (Figure 3 (e)-(i)). In the *lid* setting, the additional actions Move-to-lid, Grasp-lid, Move-to-dropoff, and Release-lid (Figure 3 (a)-(d)).



(a) (b) (c)

Fig. 4: Different metrics in relation to the number of training videos for the ProtoNet alone and with the HSMM, the sequence accuracy (a) describes how well the model learns the segment order (higher is better), the average F1-score (b) describes how well the classes are aligned (higher is better), and the average Edit-Distance (c) indicating how many edits have to be performed in order match the sequence (lower is better).



(a) Pre-training (b) Zero-Shot (c) Three-Shot

Fig. 5: Sample segmentations produced by the ProtoNet and the ProtoNet with the HSMM in comparison to the ground truth for the pre-training dataset in a supervised learning scenario (a), as well as for the main dataset in a Zero-Shot (b) and a Three-Shot Learning scenario (c). The clusters for Zero-Shot Learning have been remapped to match the ground truth.

2) Features: We use a 66-dimensional feature space consisting of automatic and handcrafted features to train the model. In order to extract those features from the RGB-D videos, we use MediaPipe Hand [44] to automatically track 25 hand landmarks and obtain 3D data for each of those landmarks. Additionally, AR-Tags are used to capture the position and the rotation data of task-relevant objects such as the trash, trash can, and lid. Based on the extracted object and hand positions, different features are calculated. These include velocities and the distance between index finger and thumb, object-object and hand-object distances. The handcrafted features are designed to provide additional information and make the features easier to handle and more invariant to changes in e.g. the camera position. In addition, models using raw data as input (e.g. transformers) come with a significantly higher computational cost, which could hinder interactivity. The incorporation of automatic and handcrafted features proved to be an effective alternative.

B. Experiment Setup

We perform an ablation study on the dataset used for pre-training and the Zero-Shot and the Few-Shot task with the added segment labels. More specifically, we evaluate the Three-Shot Learning task of the ProtoNet on its own and in combination with the HSMM and compare it against the two baselines. Our model and all baselines are first pre-trained on the *simple* task in a fully supervised setting and then trained on the *lid* task in a Zero- and Few-Shot setting.

1) Model Details: The encoding function $f_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-w:t})$ of the ProtoNet consists of 3 layers of bidirectional Gated Recurrent Units (GRU) [7] each with a hidden size 10 followed by a Fully Connected Network layer that maps the output to a 10-dimensional encoding space. A sliding window size of 60 and a dropout of 0.5 was used. The network was trained with a batch size of 70 for 150 Epochs during pre-training and 40 Epochs during each Few-Shot iteration.

2) Baselines: The first baseline is a simple three-layer Long-Short-Term-Memory (LSTM) [21] model and the second baseline is the UVAST model proposed by Behrmann et al. [2], which utilizes a Transformer seq2seq model that predicts the sequence order and the duration separately.

3) Ablation Study: We evaluate all models on the accuracy, the Macro F1-score, the Macro IoU, the Sequence Accuracy, and the Edit-Distance. Each experiment was performed five times with different random seeds. We report

	Accuracy (↑)	Macro F1-Score (↑)	Macro IoU (↑)	Sequence Accuracy (↑)	Edit-Distance (↓)
ProtoNet	$87.856 \pm 1.351\%$	$82.505 \pm 1.788\%$	$72.136 \pm 2.314\%$	$97.143 \pm 5.714\%$	0.057 ± 0.114
ProtoNet + HSMM	$88.685 \pm 1.479\%$	$81.165 \pm 1.890\%$	$72.581 \pm 2.052\%$	$\mathbf{100.000 \pm 0.000\%}$	$\mathbf{0.000 \pm 0.000}$
simple LSTM	$77.297 \pm 3.549\%$	$62.799 \pm 8.231\%$	$50.573 \pm 7.171\%$	$0.000 \pm 0.000\%$	4.000 ± 0.000
UVAST [2]	$\mathbf{91.027 \pm 0.574\%}$	$\mathbf{85.939 \pm 0.897\%}$	$\mathbf{78.172 \pm 1.286\%}$	$91.429 \pm 6.999\%$	0.086 ± 0.070

TABLE I: Performances measured on the *simple* dataset for our proposed model with and without HSMM smoothing as well as for the two baselines. We report the average values and the standard deviation of each metric. (↑ - higher is better, ↓ - lower is better)

n	Model	Accuracy (↑)	Macro F1-Score (↑)	Macro IoU (↑)	Sequence Accuracy (↑)	Edit-Distance (↓)
zero	ProtoNet	$12.069 \pm 7.417\%$	$9.294 \pm 4.211\%$	$5.596 \pm 2.794\%$	$0.000 \pm 0.000\%$	8.493 ± 1.301
	ProtoNet + HSMM	$\mathbf{36.860 \pm 5.697\%}$	$\mathbf{25.255 \pm 4.737\%}$	$\mathbf{18.087 \pm 3.718\%}$	$0.000 \pm 0.000\%$	$\mathbf{6.191 \pm 0.848}$
three	Simple LSTM	$66.551 \pm 6.499\%$	$42.699 \pm 4.518\%$	$32.585 \pm 4.603\%$	$0.000 \pm 0.000\%$	8.000 ± 0.000
	ProtoNet	$67.993 \pm 2.898\%$	$57.016 \pm 2.581\%$	$43.585 \pm 2.268\%$	$25.674 \pm 7.232\%$	2.036 ± 0.405
	ProtoNet + HSMM	$73.586 \pm 2.950\%$	$59.290 \pm 3.046\%$	$49.033 \pm 2.927\%$	$\mathbf{55.254 \pm 14.755\%}$	$\mathbf{0.877 \pm 0.310}$
	UVAST [2]	$\mathbf{82.917 \pm 1.363\%}$	$\mathbf{71.385 \pm 1.538\%}$	$\mathbf{61.585 \pm 1.922\%}$	$50.551 \pm 19.075\%$	1.025 ± 0.475

TABLE II: Performance measured on the *lid* task for our proposed model with and without HSMM smoothing and for the two baselines. The first column indicates the number of videos used for training (n-shot). All models were pre-trained on the *simple* task. We report the mean and standard deviation of each metric. (↑ - higher is better, ↓ - lower is better)

the average metric performance with the standard deviation in Tables I and II. In addition, we visualize some video segmentation examples of each scenario in Figure 5. The Zero- and Few-Shot performance is evaluated by performing five-fold on the dataset and sub-sampling n videos from the training fold for $n = 1 \dots 11$. The separate models from the $k-1$th iteration are refined at the kth iteration to mimic the iterative workflow of the user interface. Since the Zero-Shot classification produces labels w.r.t. the learned clusters, these are automatically remapped to the ground truth labels with the most overlap before computing any metric. This step would usually be performed by the human annotator using the interactive user interface alongside the correction of the initial prediction. To analyze the models performance dependent on the amount of training videos, Figures 4a and 4b visualize the average performance and standard deviation of the folds for each n for the sequence accuracy, the macro F1-score, and the Edit-Distance.

We determine the quality of the models using a set of different metrics. First of all, we compute the average frame-wise accuracy. This metric on its own, however, is not particularly strong as feature imbalances skew it towards well-represented segments. Therefore, we additionally compute the macro F1-score and the macro IoU (intersection over union). The macro F1-score

$$\mathrm{F1}_{\text{macro}}(\hat{\boldsymbol{y}}, \boldsymbol{y}) = \frac{1}{K} \sum_{k=1}^{K} \mathrm{F1}_k(\hat{\boldsymbol{y}}, \boldsymbol{y})$$

is computed as the class-wise mean of the F1-score, where

$$\mathrm{F1}_k(\hat{\boldsymbol{y}}, \boldsymbol{y}) = \frac{2 \cdot \#\mathrm{TP}}{2 \cdot \#\mathrm{TP} + \#\mathrm{FP} + \#\mathrm{FN}}$$

is the F1-score with respect to class k. Here #TP, #FP, and #FN denote the number of true positives, false positives, and false negatives, respectively.

Similarly, the macro IoU

$$\mathrm{IoU}_{\text{macro}}(\hat{\boldsymbol{y}}, \boldsymbol{y}) = \frac{1}{K} \sum_{k=1}^{K} \mathrm{IoU}_k(\hat{\boldsymbol{y}}, \boldsymbol{y})$$

is the class-wise mean of the IoU, with

$$\mathrm{IoU}_k(\hat{\boldsymbol{y}}, \boldsymbol{y}) = \frac{|\boldsymbol{I}_k|}{|\boldsymbol{U}_k|} = \frac{|\{t | \hat{y}_t = y_t = k\}|}{|\{t | \hat{y}_t = k \vee y_t = k\}|},$$

where $\hat{y}_t \in \hat{\boldsymbol{y}}, y_t \in \boldsymbol{y}$. Here, $\boldsymbol{I}_k$ is the intersection of the prediction and the ground truth on k and $\boldsymbol{U}_k$ is the union thereof. Lastly, we compute the average Levenshtein Edit-Distance, which is the minimum number of changes (insertions, deletions, or replacements) needed to convert the predicted sequence into the ground truth. Therefore, the Levenshtein Edit-Distance roughly corresponds to the number of changes the user needs to make. Additionally, we refer to the fraction of videos with correctly ordered sequences as the sequence accuracy. Both of these metrics are computed w.r.t. the resulting segment order and do not take the alignment of the start and end points w.r.t the ground truth into account.

C. Results

When inspecting the pre-training metrics reported in Table I, one can see that the ProtoNet is able to outperform the simple LSTM across all metrics with and without the added HSMM smoothing. However, the more complex UVAST model [2] outperforms our model for the metrics, which take alignment into account. This indicates that while our model is able to learn the correct sequence of segments, there is still room for improvement when it comes to aligning them correctly. This discrepancy also becomes obvious when inspecting the Few-Shot results presented in Table II. While we still outperform the Few-Shot LSTM, the UVAST model is still able to predict the segmentations more accurately

when it comes to alignment. However, our method reaches a higher sequence accuracy and lower edit distance compared to both baselines. Because of that, the amount of correction needed by the user is lowered, making the labeling process quicker and easier. Moreover, further usage of the resulting segmentation e.g. in Reinforcement Learning may benefit more from an accurate sequence ordering than from a well-aligned but over-segmented prediction. In Figures 5a and 5c sample segmentations for ProtoNet and ProtoNet combined with HSMM are visualized in comparison to the ground truth. The order is accurately predicted for the most parts, but there are noticeable offsets in the start and endpoints of some segments reflecting the aforementioned lower accuracy.

For the Zero-Shot Learning scenario, there is plenty of room for improvement. Furthermore, we observe a high standard deviation in the results (Table II). Although the *lid* task contains some of the actions of the *simple* task, the model does not seem to adapt well to the new classes during Zero-Shot as is shown in Figure 5b. In this case, as well as for Few-Shot Learning, the HSMM improves the results a lot (see Table II and Figures 4a, 4b, and 4c).

Lastly, our analysis regarding the effect of training videos (see Figure 4a, 4b, and 4c) shows that the performance starts to converge at around six training videos.

V. Conclusion

In this paper, we proposed I^3: Interactive Iterative Improvement for Few-Shot Action Segmentation from limited labeled data obtained via user-annotated sequences. We do so using a pre-trained Action Segmentation model that is interactively improved in a Few-Shot manner by an end user via a webinterface. This framework was designed as a basis for Action Segmentation with non-experts end users in mind.

We evaluated our method on human demonstrations of a trash disposal task and compared the performance against two baselines. While the accuracy of our model is lower compared to the UVAST baseline, our model reaches a higher sequence accuracy and lower Edit-Distances which is linked to a lower number of required user edits and makes it suitable for use cases that require accurate sequence orderings such as in robot skill learning from human demonstrations.

In future work, we want to investigate ways to improve the overall performance by adding some regularization from related works to the ProtoNet to ensure smoother segment transitions or exploring different clustering methods or an ensemble of methods. In addition, this approach could be enhanced by better incorporating the HSMM with the ProtoNet thereby learning temporally coherent encodings. The HSMM could also be improved by adding constraints regarding the order of transitions, similar to left-to-right HSMMs [45]. Moreover, we want to explore a more diverse set of tasks without such a high similarity, as this can influence the pre-training of the Zero-/Few-Shot Learning. In order to enhance the comparability of future investigations, it is imperative to incorporate a broader spectrum of baselines for more comprehensive comparative analyses. Finally, performing a user study to evaluate the fit of the proposed framework with principles for interactive machine learning [10] is something we would explore in our future work, along with learning robotic controllers to imitate the segmented actions.

References

[1] M. V. Balakuntala et al. "Extending policy from one-shot learning through coaching". In: *IEEE International Conference on Robot and Human Interactive Communication (RO-MAN)*. 2019.

[2] N. Behrmann et al. "Unified fully and timestamp supervised temporal action segmentation via sequence to sequence translation". In: *European Conference on Computer Vision*. 2022.

[3] Y. Y. Boykov and M.-P. Jolly. "Interactive graph cuts for optimal boundary & region segmentation of objects in ND images". In: *IEEE International Conference on Computer Vision (ICCV)*. 2001.

[4] E. Brachmann et al. "Learning 6D Object Pose Estimation Using 3D Object Coordinates". In: *Proceedings of the European Conference on Computer Vision*. Ed. by D. Fleet et al. 2014.

[5] S. Calinon. "A tutorial on task-parameterized movement learning and retrieval". In: *Intelligent Service Robotics* (2016).

[6] J. Carreira and A. Zisserman. "Quo vadis, action recognition? a new model and the kinetics dataset". In: *IEEE Conference on Computer Vision and Pattern Recognition*. 2017.

[7] K. Cho et al. "On the properties of neural machine translation: Encoder-decoder approaches". In: *Workshop on Syntax, Semantics and Structure in Statistical Translation*. Association for Computational Linguistics, 2014.

[8] H. Dang et al. "How to Prompt? Opportunities and Challenges of Zero-and Few-Shot Learning for Human-AI Interaction in Creative Applications of Generative Models". In: *Generative AI and HCI Workshop at CHI 2022* (2022).

[9] G. De Rossi et al. "A first evaluation of a multi-modal learning system to control surgical assistant robots via action segmentation". In: *IEEE Transactions on Medical Robotics and Bionics* (2021).

[10] J. J. Dudley and P. O. Kristensson. "A review of user interface design for interactive machine learning". In: *ACM Transactions on Interactive Intelligent Systems (TiiS)* (2018).

[11] A. Fathi, X. Ren, and J. M. Rehg. "Learning to recognize objects in egocentric activities". In: *IEEE Conference on Computer Vision and Pattern Recognition*. 2011.

[12] L. Fei-Fei, R. Fergus, and P. Perona. "One-shot learning of object categories". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2006).

[13] C. Feichtenhofer, A. Pinz, and R. P. Wildes. "Temporal residual networks for dynamic scene recognition". In: *IEEE Conference on Computer Vision and Pattern Recognition*. 2017.

[14] R. Feng et al. “Interactive few-shot learning: Limited supervision, better medical image segmentation”. In: *IEEE Transactions on Medical Imaging* (2021).

[15] E. Ferreira et al. “Adversarial bandit for online interactive active learning of zero-shot spoken language understanding”. In: *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2016.

[16] M. Fink. “Object classification from a single example utilizing class relevance metrics”. In: *Advances in Neural Information Processing Systems* (2004).

[17] Y. Gao et al. “JHU-ISI Gesture and Skill Assessment Working Set (JIGSAWS): A Surgical Activity Dataset for Human Motion Modeling”. In: *Modeling and Monitoring of Computer Assisted Interventions (M2CAI) – MICCAI Workshop* (2014).

[18] R. Ghoddoosian et al. “Weakly-supervised online action segmentation in multi-view instructional videos”. In: *IEEE Conference on Computer Vision and Pattern Recognition*. 2022.

[19] K. He et al. “Mask R-CNN”. In: *Proceedings of the International Conference on Computer Vision*. 2017.

[20] D. J. Hejna III and D. Sadigh. “Few-shot preference learning for human-in-the-loop RL”. In: *Conference on Robot Learning*. PMLR. 2023.

[21] S. Hochreiter and J. Schmidhuber. “Long short-term memory”. In: *Neural Computation* (1997).

[22] S. Jain, S. Munukutla, and D. Held. “Few-Shot Point Cloud Region Annotation with Human in the Loop”. In: *ICML Workshop on Human in the Loop Learning (HILL)*. 2019.

[23] H. Kuehne, A. Arslan, and T. Serre. “The language of actions: Recovering the syntax and semantics of goal-directed human activities”. In: *IEEE Conference on Computer Vision and Pattern Recognition*. 2014.

[24] H. Kuehne, A. Richard, and J. Gall. “A Hybrid RNN-HMM Approach for Weakly Supervised Temporal Action Segmentation”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2020).

[25] A. Kukleva et al. “Unsupervised learning of action classes with continuous temporal embedding”. In: *IEEE Conference on Computer Vision and Pattern Recognition*. 2019.

[26] J. Li, P. Lei, and S. Todorovic. “Weakly supervised energy-based learning for action segmentation”. In: *International Conference on Computer Vision*. 2019.

[27] S. Oshikawa et al. “Interaction modeling based on segmenting two persons motions using coupled GP-HSMM”. In: *IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)*. IEEE. 2018.

[28] E. Pignat and S. Calinon. “Learning adaptive dressing assistance from human demonstration”. In: *Robotics and Autonomous Systems* (2017).

[29] K. Rakelly et al. “Few-shot segmentation propagation with guided networks”. In: *arXiv preprint arXiv:1806.07373* (2018).

[30] A. Richard et al. “Neuralnetwork-viterbi: A framework for weakly supervised video learning”. In: *IEEE Conference on Computer Vision and Pattern Recognition*. 2018.

[31] S. Sarfraz et al. “Temporally-weighted hierarchical clustering for unsupervised action segmentation”. In: *IEEE Conference on Computer Vision and Pattern Recognition*. 2021.

[32] D. Singhania, R. Rahaman, and A. Yao. “Coarse to fine multi-resolution temporal convolutional network”. In: *arXiv preprint arXiv:2105.10859* (2021).

[33] J. Snell, K. Swersky, and R. Zemel. “Prototypical networks for few-shot learning”. In: *Advances in Neural Information Processing Systems* (2017).

[34] G. Stavropoulos et al. “Automatic Action Recognition for Assistive Robots to Support MCI Patients at Home”. In: *International Conference on PErvasive Technologies Related to Assistive Environments (PETRA)*. Association for Computing Machinery, 2017.

[35] M. J. Trimpl et al. “Beyond automatic medical image segmentation—the spectrum between fully manual and fully automatic delineation”. In: *Physics in Medicine & Biology* (2022).

[36] S. Wan et al. “Human-in-the-loop low-shot learning”. In: *IEEE Transactions on Neural Networks and Learning Systems* (2020).

[37] M. Wang et al. “Learning functional sections in medical conversations: iterative pseudo-labeling and human-in-the-loop approach”. In: *arXiv preprint arXiv:2210.02658* (2022).

[38] P. Wang et al. “DemoGrasp: Few-shot learning for robotic grasping with human demonstration”. In: *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2021.

[39] Y. Wang et al. “Generalizing from a few examples: A survey on few-shot learning”. In: *ACM Computing Surveys* (2020).

[40] X. Wu et al. “A survey of human-in-the-loop for machine learning”. In: *Future Generation Computer Systems* (2022).

[41] C. Xiao et al. “Self-Supervised Few-Shot Time-series Segmentation for Activity Recognition”. In: *IEEE Transactions on Mobile Computing* (2022).

[42] X. Xu et al. “Enabling hand gesture customization on wrist-worn devices”. In: *2022 CHI Conference on Human Factors in Computing Systems*. 2022.

[43] S.-Z. Yu. “Hidden semi-Markov models”. In: *Artificial Intelligence* (2010).

[44] F. Zhang et al. “Mediapipe hands: On-device real-time hand tracking”. In: *arXiv preprint arXiv:2006.10214* (2020).

[45] X. Zhu et al. “A comparative study of mixture-Gaussian VQ, ergodic HMMs and left-to-right HMMs for speaker recognition”. In: *International Conference on Speech, Image Processing and Neural Networks*. 1994.