Weakly Supervised Energy-Based Learning for Action Segmentation

Jun Li
Oregon State University
liju2@oregonstate.edu

Peng Lei*
Amazon.com Services, Inc.
leipeng@amazon.com

Sinisa Todorovic
Oregon State University
sinisa@oregonstate.edu

Abstract

This paper is about labeling video frames with action classes under weak supervision in training, where we have access to a temporal ordering of actions, but their start and end frames in training videos are unknown. Following prior work, we use an HMM grounded on a Gated Recurrent Unit (GRU) for frame labeling. Our key contribution is a new constrained discriminative forward loss (CDFL) that we use for training the HMM and GRU under weak supervision. While prior work typically estimates the loss on a single, inferred video segmentation, our CDFL discriminates between the energy of all valid and invalid frame labelings of a training video. A valid frame labeling satisfies the ground-truth temporal ordering of actions, whereas an invalid one violates the ground truth. We specify an efficient recursive algorithm for computing the CDFL in terms of the logadd function of the segmentation energy. Our evaluation on action segmentation and alignment gives superior results to those of the state of the art on the benchmark Breakfast Action, Hollywood Extended, and 50Salads datasets.†

1. Introduction

This paper presents an approach to weakly supervised action segmentation by labeling video frames with action classes. Weak supervision means that in training our approach has access only to the temporal ordering of actions, but their ground-truth start and end frames are not provided. This is an important problem with a wide range of applications, since the more common fully supervised action segmentation typically requires expensive manual annotations of action occurrences in every video frame.

Our fundamental challenge is that the set of all possible segmentations of a training video may consist of multiple distinct *valid* segmentations that satisfy the provided ground-truth ordering of actions, along with *invalid* segmentations that violate the ground truth. It is not clear how to estimate loss (and subsequently train the segmenter) over multiple valid segmentations.

Motivation: Prior work [8, 12, 20, 7, 22] typically uses a temporal model (e.g., deep neural network, or HMM) to infer a *single*, valid, optimal video segmentation, and takes this inference result as a pseudo ground truth for estimating the incurred loss. However, a particular training video may exhibit a significant variation (not yet captured by the model along the course of training), which may negatively affect estimation of the pseudo ground truth, such that the inferred action segmentation is significantly different from the true one. In turn, the loss estimated on the incorrect pseudo ground truth may corrupt training by reducing, instead of maximizing, the discriminative margin between the ground truth and other valid segmentations. In this paper, we seek to alleviate these issues.

Contributions: Prior work shows that a statistical language model is useful for weakly supervised learning and modeling of video sequences [17, 9, 19, 22, 3]. Following [22], we also adopt a Hidden Markov Model (HMM) grounded on a Gated Recurrent Unit (GRU) [4] for labeling video frames. The major difference is that we do not generate a unique pseudo ground truth for training. Instead, we efficiently account for all candidate segmentations of a training video when estimating the loss. To this end, we formulate a new Constrained Discriminative Forward Loss (CDFL) as a difference between the energy of valid and invalid candidate video segmentations. In comparison with prior work, the CDFL improves robustness of our training, because minimizing the CDFL amounts to maximizing the discrimination margin between candidate segmentations that satisfy and violate ground truth, whereas prior work solely optimizes a score of the inferred single valid segmentation. Robustness of training is further improved when the CDFL takes into account only hard invalid segmentations whose edge energy is lower than that of valid ones. Along with the new CDFL formulation, our key contribution is a new recursive algorithm for efficiently estimating the CDFL in terms of the *logadd* function of the segmentation energy.

Our Approach: Fig. 1 shows an overview of our weakly supervised training of the HMM with GRU that consists of

*The work was done at the Oregon State University before Peng Lei joined Amazon. †The code is available at https://github.com/JunLi-Galios/CDFL.

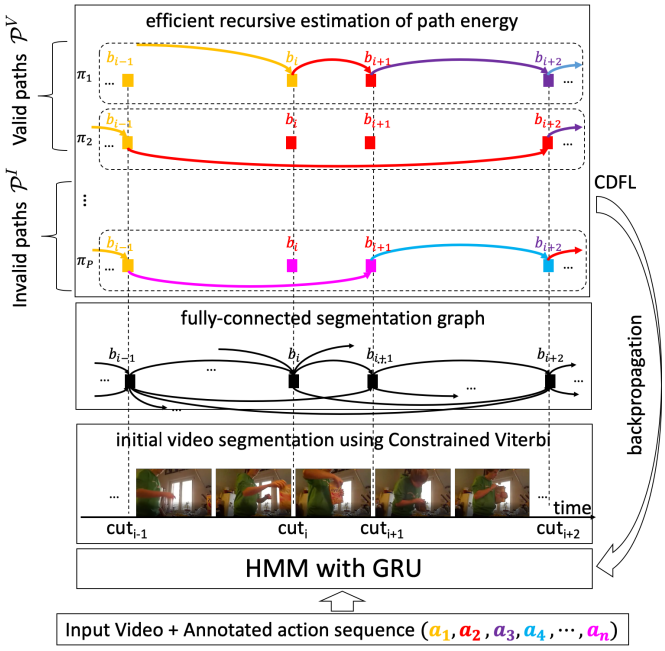

Figure 1. **Our weakly supervised training**: For a training video, we first estimate candidate segmentation cuts using a Hidden Markov Model (HMM) grounded on a Gated Recurrent Unit (GRU), and then build a fully connected segmentation graph whose paths represent candidate action segmentations (colors mark different action classes along the paths). Then, we efficiently compute the Constrained Discriminative Forward Loss (CDFL) in terms of accumulated energy of all valid and invalid paths in the graph for our end-to-end training. (best seen in color)

two steps. In the first step, we run a constrained Viterbi algorithm for HMM inference on a given training video so the resulting segmentation is valid. This initial video segmentation is used for efficiently building a fully connected segmentation graph aimed at representing alternative candidate segmentations. In this graph, nodes represent segmentation cuts of the initially inferred segmentation – i.e., video frames where one action ends and a subsequent one starts – and edges represent video segments between every two temporally ordered cuts. For improving action boundary detection, we further augment the initial set of nodes with video frames that are in a vicinity of every cut, as well as the initial set of edges with corresponding temporal links between the added nodes. Directed paths of such a fully connected graph explicitly represent many candidate action segmentations, beyond the initial HMM's inference.

The second step of our training efficiently computes a total energy score of frame labeling along *all paths* in the segmentation graph. Efficiency comes from our novel recursive estimation of the segmentation energy, where we exploit the accumulative property of the *logadd* function. A difference of the accumulated energy of action labeling along the valid and invalid paths is used to compute the CDFL. In this paper, we also consider several other loss formulations expressed in terms of the energy of valid and invalid paths. The loss is then used for training HMM parameters and back-propagated to the GRU for end-to-end training.

For inference on a test video, as in the first step of our training, we use a constrained Viterbi algorithm to perform the HMM inference which will satisfy at least one action sequence seen in training. Then, we use this initial video segmentation as an anchor for building the segmentation graph that comprises paths with finer action boundaries. Our output is the MAP path in the graph.

For evaluation, we consider the tasks of action segmentation and action alignment, where the latter provides additional information on the temporal ordering of actions in the test video. For both tasks on the Breakfast Action dataset [10], Hollywood Extended dataset [1], and 50-Salads dataset [24], we outperform the state of the art.

In the following, Sec. 2 reviews related work, Sec. 3 formulates our HMM and Constrained Viterbi for action segmentation, Sec. 4 describes how we construct the segmentation graph, Sec. 5 specifies our CDFL and related loss functions, and Sec. 6 presents our evaluation.

2. Related Work

This section reviews closely related work on weakly supervised action segmentation and Graph Transformer Networks. While a review of fully supervised action segmentation [25, 14, 18, 16] is beyond our scope, it is worth mentioning that our approach uses the same recurrent deep models for frame labeling as in [23, 25, 6]. Also, our approach is motivated by [11, 19] which integrate HMMs and modeling of action length priors within a deep learning architecture.

Weakly supervised action segmentation has recently made much progress [24, 10, 20, 7, 22]. For example, Extended Connectionist Temporal Classification (ECTC) addresses action alignment under the constraint of being consistent with frame-to-frame visual similarity [8]. Also, action segmentation has been addressed with a convex relaxation of discriminative clustering, and efficiently solved with the conditional gradient (Frank-Wolfe) algorithm [1]. Other approaches use a local action model and a global temporal alignment model that are alternatively trained [12, 20]. Some methods initially predict a video segmentation with a temporal convolutional network, and then iteratively refine the action boundaries [7]. Other approaches first generate pseudo-ground-truth labels for all video frames, e.g., with the Viterbi algorithm [22], and then train a classifier on these frame labels by minimizing the standard cross entropy loss. Finally, [21] addresses a different weakly supervised setting from ours when the ground truth provides only a set of actions present without their temporal ordering .

All these approaches base their learning and prediction on estimating a penalty or probability of labeling individ-

ual frames. In contrast, we use an energy-based framework with the following differences. First, in training, we minimize the total energy of valid paths in the segmentation graph rather than optimize labeling probabilities of each frame. Second, instead of considering a single optimal valid path in the segmentation graph, we specify a loss function in terms of *all* valid paths. Hence, the Viterbi-initialized training on pseudo-labels of frames [22] represents a special case of our training done only for one valid path. In addition, our loss enforces discriminative training by accounting for invalid paths in the segmentation graph. Unlike [3] that randomly selects invalid paths, we efficiently account for all hard invalid paths in training. Finally, our training is not iterative as in [12, 20], and does not require iterative refinement of action boundaries as in [7].

Our CDFL extends the loss used for training of the Graph Transformer Network (GTN) [15, 13, 2, 5]. To the best of our knowledge, the GTN has been used only for text parsing, and never for action segmentation. In comparison with the GTN training, we significantly reduce complexity by building the video's segmentation graph. Also, while the loss used for training the GTN accounts for both valid and invalid text parses, it cannot handle the special case when valid parses have lower scores than invalid ones. In contrast, our CDFL effectively accounts for the energy of valid and invalid paths, even when valid paths have significantly lower energy than invalid paths in the segmentation graph.

3. Our Model for Action Segmentation

Problem Setup: For each training video of length T, we are given unsupervised frame-level features, $\boldsymbol{x}_{1:T} = [x_1, x_2, ..., x_T]$, and the ground-truth ordering of action classes $\boldsymbol{a}_{1:N} = [a_1, a_2, ..., a_N]$, also referred to as the transcript. N is the length of the annotation sequence, and a_n is nth action class in $\boldsymbol{a}_{1:N}$ that belongs to the set of K action classes, $a_n \in \mathcal{A} = \{1, 2, ..., K\}$. Note that T and N may vary across the training set, and that there may be more than one occurrences of the same action class spread out in $\boldsymbol{a}_{1:N}$ (but of course $a_n \neq a_{n+1}$).

In inference, given frame features $\boldsymbol{x}_{1:T}$ of a video, our goal is to find an optimal segmentation $(\hat{\boldsymbol{a}}_{1:\hat{N}}, \hat{\boldsymbol{l}}_{1:\hat{N}})$, where $\hat{N}$ is the predicted length of the action sequence, and $\hat{\boldsymbol{l}}_{1:\hat{N}} = [\hat{l}_1, \hat{l}_2, \cdots, \hat{l}_{\hat{N}}]$ includes the predicted number of video frames $\hat{l}_n$ occupied by the predicted action $\hat{a}_n$.

The Model: We use an HMM to model the posterior distribution of a video segmentation $(\boldsymbol{a}_{1:N}, \boldsymbol{l}_{1:N})$ given $\boldsymbol{x}_{1:T}$ as

$$\begin{aligned} &p(\boldsymbol{a}_{1:N}, \boldsymbol{l}_{1:N}|\boldsymbol{x}_{1:T}) \\ &\quad \propto p(\boldsymbol{x}_{1:T}|\boldsymbol{a}_{1:N}, \boldsymbol{l}_{1:N}) p(\boldsymbol{l}_{1:N}|\boldsymbol{a}_{1:N}) p(\boldsymbol{a}_{1:N}), \\ &\quad = \left(\prod_{t=1}^{T} p(x_t|a_{n(t)}) \right) \left(\prod_{n=1}^{N} p(l_n|a_n) \right) p(\boldsymbol{a}_{1:N}). \end{aligned} \tag{1}$$

In (1), the likelihood $p(x_t|a)$ is estimated as

$$p(x_t|a) \propto \frac{p(a|x_t)}{p(a)}, \tag{2}$$

where $p(a|x_t)$ is the GRU's softmax score for action $a \in \mathcal{A}$ at frame t, and the prior distribution of action classes $p(a)$ is an normalized frame frequency of action occurrences in the training dataset. The likelihood of action length is modeled as a class-dependent Poisson distribution

$$p(l|a) = \frac{\lambda_a^l}{l!} e^{-\lambda_a}, \tag{3}$$

where λ_a is the mean length for class $a \in \mathcal{A}$. Finally, the joint prior $p(\boldsymbol{a}_{1:N})$ is a constant if the transcript $\boldsymbol{a}_{1:N}$ exists in the training set; otherwise, $p(\boldsymbol{a}_{1:N}) = 0$. The same modeling formulation was well-motivated and used in state of the art [22].

Constrained Viterbi Algorithm: Given a training video, we first find an optimal valid action segmentation $(\hat{\boldsymbol{a}}_{1:\hat{N}}, \hat{\boldsymbol{l}}_{1:\hat{N}})$ by maximizing (1) with a constrained Viterbi algorithm, which ensures that $\hat{\boldsymbol{a}}_{1:\hat{N}}$ is equal to the annotated transcript, $\hat{\boldsymbol{a}}_{1:\hat{N}} = \boldsymbol{a}_{1:N}$. Similarly, for inference on a test video, we first perform the constrained Viterbi algorithm against all transcripts $\{\boldsymbol{a}_{1:N}\}$ seen in training, i.e., ensure that the predicted $\hat{\boldsymbol{a}}_{1:\hat{N}}$ has been seen at least once in training. Thus, the initial step of our inference on a training or test video is the same as in [22].

Our key difference from [22], is that we use the initial $(\hat{\boldsymbol{a}}_{1:\hat{N}}, \hat{\boldsymbol{l}}_{1:\hat{N}})$ to efficiently build a fully connected segmentation graph of the video, as explained in Sec. 4. Importantly, in training, the segmentation graph is not constructed to find a more optimal video segmentation that improves upon the initial prediction. Instead, the graph is used to efficiently account for all valid and invalid segmentations.

Given a video $\boldsymbol{x}_{1:T}$ and a transcript $\boldsymbol{a}_{1:N}$, the constrained Viterbi algorithm recursively maximizes the posterior in (1) such that the first n action labels of the transcript $\boldsymbol{a}_{1:n} = [a_1, ..., a_n] \subseteq \boldsymbol{a}_{1:N}$ are respected at time t:

$$\begin{aligned} p(\boldsymbol{a}_{1:n}, \hat{\boldsymbol{l}}_{1:n}|\boldsymbol{x}_{1:t}) = \max_{t',\, t'<t} \Bigg\{ & p(\boldsymbol{a}_{1:n-1}, \hat{\boldsymbol{l}}_{1:n-1}|\boldsymbol{x}_{1:t'}) \\ & \cdot \left(\prod_{s=t'}^{t} p(x_s|a_{n(s)}) \right) \cdot p(l_n|a_n) \cdot p(\boldsymbol{a}_{1:n}) \Bigg\}, \end{aligned} \tag{4}$$

where $l_n = t - t'$. We set $p(\cdot|\boldsymbol{x}_{1:0}) = 1$, and $p(\boldsymbol{a}_{1:n}) = \kappa$, where $\kappa > 0$ is a constant.

4. Constructing the Segmentation Graph

Given a video $\boldsymbol{x}_{1:T}$, we first run the constrained Viterbi algorithm to obtain an initial video segmentation $(\hat{\boldsymbol{a}}_{1:\hat{N}}, \hat{\boldsymbol{l}}_{1:\hat{N}})$. For simplicity, in the following, we ignore the symbol $\hat{}$. This initial segmentation is characterized by

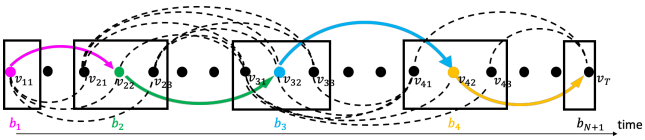

Figure 2. **Building the segmentation graph** G (best seen in color). The initial nodes of G represent segmentation cuts b_n obtained in the Constrained Viterbi (the predicted action classes are marked with different colors). Each b_n generates additional vertices $b_n = \{v_{ns}\}$ representing neighboring video frames within a window centered at b_n (the black rectangles), and corresponding new edges $(v_{ns}, v_{n's'})$ (the dashed lines) between all temporally ordered pairs of vertices in G. For clarity, we show only a few edges. G has exponential many paths, each representing a candidate action segmentation.

$N+1$ cuts, $\boldsymbol{b}_{1:N+1} = [b_1, \ldots, b_{N+1}]$, i.e., video frames where previous action ends and the next one starts including the very first frame b_1 and last frame b_{N+1} at time T.

We use these cuts to anchor our construction of the fully connected segmentation graph, $G = (\mathcal{V}, \mathcal{E}, \mathcal{W})$, where $\mathcal{V} = \{\boldsymbol{b}_{1:N+1}\}$ is the set of nodes, $\mathcal{E}$ is the set of directed edges linking every two temporally ordered nodes, and $\mathcal{W}$ are the corresponding edge weights.

Some of the estimated cuts in $\boldsymbol{b}_{1:N+1}$ may be false positives or may not exactly coincide with the true cuts. To improve action boundary detection, we augment the initial $\mathcal{V}$ with nodes representing neighboring video frames of each cut b_n within a temporal window of length Δ centered at b_n, as illustrated in Fig. 2. For the first and last frames, we set $\Delta = 1$. Thus, each b_n can be viewed as a hyper-node comprising additional vertices in G, $\mathcal{V} = \{b_n = \{v_{n1}, \cdots, v_{ni}, \cdots, v_{n\Delta}\} : n = 1, \ldots, N+1\}$, and accordingly additional edges $\mathcal{E} = \{(v_{ni}, v_{n'i'}) : n \leq n',\ i < i'\}$. In the following, we simplify notation for vertices $v_{ni} \rightarrow v_i \in \mathcal{V}$, and edges $(v_{ni}, v_{n'i'}) \rightarrow e_{ii'} = (v_i, v_{i'})$.

Each edge $e_{ii'}$ is assigned a weight vector $\boldsymbol{w}_{ii'} = [w_{ii'}(a)]$, where $w_{ii'}(a)$ is defined as the energy of labeling the video segment $(v_i, v_{i'})$ with action class $a \in \mathcal{A}$:

$$w_{ii'}(a) = \sum_{t \in (v_i, v_{i'})} -\log p(a|x_t), \tag{5}$$

where $p(a|x_t)$ is the GRU's softmax score for action a at frame t.

G comprises exponentially many directed paths $\mathcal{P} = \{\pi\}$, where each π represents a particular video segmentation. In each π, every edge $e_{ii'}$ gets assigned only one action class $a^{\pi}_{ii'} \in \mathcal{A}$. Thus, the very same edge with K different class assignments belongs to K distinct paths in $\mathcal{P}$. We compute the energy of a path as

$$E_\pi = \sum_{e_{ii'} \in \pi} w_{ii'}(a^{\pi}_{ii'}). \tag{6}$$

A subset of valid paths $\mathcal{P}^V \subset \mathcal{P}$ satisfies the given transcript. The other paths are invalid, $\mathcal{P}^I = \mathcal{P} \setminus \mathcal{P}^V$.

In the next section, we explain how to efficiently compute a total energy score of the exponentially many paths in $\mathcal{P}$ for estimating our loss in training.

5. Constrained Discriminative Forward Loss

In this paper, we study *three* distinct loss functions, defined in terms of a total energy score of paths in G. As there are exponentially many paths in G, our key contribution is the algorithm for efficiently estimating their total energy. Below, we specify our three loss functions ordered by their complexity. As we will show in Sec. 6, we obtain the best performance when using the CDFL in training.

5.1. Forward Loss

We define a forward loss, L_{F}, in terms of a total energy of all valid paths using the standard *logadd* function as

$$L_{\text{F}} = -\log(\sum_{\pi \in \mathcal{P}^V} \exp(-E_\pi)), \tag{7}$$

where energy of a path E_π is given by (6). As there are exponentially many paths in $\mathcal{P}^V$, we cannot directly compute L_{F} as specified in (7). Therefore, we derive a novel recursive algorithm for accumulating the energy scores of edges along multiple paths, as specified below.

We begin by defining the *logadd* function as

$$\text{logadd}(a, b) = -\log(\exp(-a) + \exp(-b)). \tag{8}$$

Note that the *logadd* function is commutative and associative, so it can be defined on a set S in a recursive manner:

$$\text{logadd}(S) = \text{logadd}(S \backslash \{x\}, x), \tag{9}$$

where x is an element in S. Therefore, the forward loss given by (7) can be expressed as

$$L_{\text{F}} = \text{logadd}(\{E_\pi : \pi \in \mathcal{P}^V\}). \tag{10}$$

Below, we simplify notation as $L_{\text{F}} = \text{logadd}(\mathcal{P}^V)$.

We recursively compute the energy score $\ell_{i'}(\boldsymbol{a}_{1:n})$ of a path that ends at node i' and covers first n labels of the ground truth $\boldsymbol{a}_{1:n} = [a_1, ..., a_n] \subseteq \boldsymbol{a}_{1:N}$ in terms of the *logadd* scores $\ell_i(\boldsymbol{a}_{1:n-1})$ of all valid paths that end at node i, $i < i'$, and cover first $n-1$ labels as

$$\ell_{i'}(\boldsymbol{a}_{1:n}) = \text{logadd}(\{\ell_i(\boldsymbol{a}_{1:n-1}) + w_{ii'}(a_n) : i < i'\}). \tag{11}$$

To prove (11), suppose that

$$\begin{aligned} \ell_i(\boldsymbol{a}_{1:n-1}) &= \text{logadd}(\{E_{\pi_i} : \pi_i \in \mathcal{P}^V\}) \\ &= -\log(\sum_{\pi_i \in \mathcal{P}^V} \exp(-E_{\pi_i})), \end{aligned} \tag{12}$$

Input: $G,\ \boldsymbol{b}_{1:N+1},\ \boldsymbol{a}_{1:N}$
Output: Forward loss $L_{\text{F}} = \ell_T(\boldsymbol{a})$

Initialization: $\ell_0(\cdot) = 0$;
for $n = 1$ **to** N **do**
  for i' in the neighborhood of b_n **do**
    $\ell_{i'}(\boldsymbol{a}_{1:n}) = \infty$;
    for i in the neighborhood of b_{n-1} **do**
      temp $= \ell_i(\boldsymbol{a}_{1:n-1}) + w_{ii'}(a_n)$;
      $\ell_{i'}(\boldsymbol{a}_{1:n}) = \text{logadd}(\ell_{i'}(\boldsymbol{a}_{1:n}), \text{temp})$;
    end
  end
end

Algorithm 1: Computing the Forward loss L_{F}.

Input: $G,\ \boldsymbol{b}_{1:N+1}$
Output: $\text{logadd}(\mathcal{P}) = \ell_T$

Initialization: $\ell_0 = 0$;
for $n = 1$ **to** N **do**
  for i' in the neighborhood of b_n **do**
    $\ell_{i'} = \infty$;
    for i in the neighborhood of b_{n-1} **do**
      for $a \in \mathcal{A}$ **do**
        $\ell_{i'} = \text{logadd}(\ell_{i'}, \ell_i + w_{ii'}(a))$;
      end
    end
  end
end

Algorithm 2: Computing the logadd score of all paths in $\mathcal{P}$, for the discriminative forward loss L_{DF}.

where π_i is a path that ends at i with a transcript of $\boldsymbol{a}_{1:n-1}$. Then, we have

$$\begin{aligned}\ell_{i'}(\boldsymbol{a}_{1:n}) &= \text{logadd}(\{\ell_i(\boldsymbol{a}_{1:n-1}) + w_{ii'}(a_n) : i < i'\}).\\ &= -\log(\sum_{i<i'}\sum_{\pi'_i \in \mathcal{P}^V} \exp(-E_{\pi_i} - w_{ii'}(a_n)))\\ &= -\log(\textstyle\sum_{\pi'_{i'} \in \mathcal{P}^V} \exp(-E_{\pi'_{i'}}))\\ &= \text{logadd}(\{E_{\pi_{i'}} : \pi_{i'} \in \mathcal{P}^V\}).\end{aligned} \tag{13}$$

where $\pi_{i'}$ is a path that ends at i' with a transcript of $\boldsymbol{a}_{1:n}$. For a training video with length T and ground-truth constraint sequence $\boldsymbol{a}_{1:N}$, we define

$$L_{\text{F}} = \ell_T(\boldsymbol{a}_{1:N}). \tag{14}$$

The recursive algorithm for computing L_{F} is presented in Alg. 1. It is worth noting that in a special case of Alg. 1, when we take only the initial segmentation cuts $\boldsymbol{b}_{1:N+1}$ as nodes of G (i.e., the window size $\Delta = 0$), the forward loss is equal to the training loss used in [22].

5.2. Discriminative Forward Loss

We also consider the Discriminative Forward Loss, L_{DF}, which extends L_{F} by additionally accounting for invalid paths in G:

$$L_{\text{DF}} = \text{logadd}(\mathcal{P}^V) - \alpha\ \text{logadd}(\mathcal{P}), \tag{15}$$

where $\text{logadd}(\mathcal{P})$ aggregates a total energy of all paths in G, and $\alpha > 0$ is a regularization factor that controls the relative importance of the valid and invalid paths for L_{DF}. Alg. 2 summarizes our recursive algorithm for computing $\text{logadd}(\mathcal{P})$ in (15), whereas Alg. 1 shows how to compute $\text{logadd}(\mathcal{P}^V)$ in (15).

One advantage of L_{DF} over L_{F} is that minimizing L_{DF} amounts to maximizing the decision margin between the valid and invalid paths. However, a potential shortcoming of L_{DF} is that valid paths might have little effect in (15). In the case, when the energy of valid paths dominates the total energy of all paths, the former gets effectively subtracted in (15), and hence has very little effect on learning.

Moreover, we observe that in some cases the back-propagation of L_{DF} is dominated by the invalid paths. This can be clearly seen from the following derivation. We compute the gradient ∇L_{DF} as

$$\begin{aligned}\nabla L_{\text{DF}} &= \nabla\text{logadd}(\mathcal{P}^V) - \alpha\ \nabla\text{logadd}(\mathcal{P}),\\ &= c_1 \textstyle\sum_{\pi\in\mathcal{P}^V} \exp(-E_\pi)\nabla E_\pi\\ &\quad -c_2 \textstyle\sum_{\pi\in\mathcal{P}^I} \exp(-E_\pi)\nabla E_\pi,\end{aligned} \tag{16}$$

where

$$\begin{aligned}c_1 &= \frac{(1-\alpha)\sum_{\pi\in\mathcal{P}^V}\exp(-E_\pi)+\sum_{\pi\in\mathcal{P}^I}\exp(-E_\pi)}{(\sum_{\pi\in\mathcal{P}^V}\exp(-E_\pi))(\sum_{\pi\in\mathcal{P}}\exp(-E_\pi))},\\ c_2 &= \frac{\alpha}{\sum_{\pi\in\mathcal{P}}\exp(-E_\pi)}.\end{aligned} \tag{17}$$

From (16)–(17), we note that in the case of $\alpha \to 1$, the backpropagation will be dominated by the invalid paths, whereas there would be no effect for invalid paths in training if $\alpha = 0$. Sec. 6 presents how different choices of α affect our performance.

In the next section, we define the constrained discriminative forward loss to address this issue.

5.3. Constrained Discriminative Forward Loss

We define the CDFL as

$$L_{\text{CDF}} = \text{logadd}(\mathcal{P}^V) - \text{logadd}(\mathcal{P}^{I_c}), \tag{18}$$

where $\mathcal{P}^{I_c}$ consists of a subset of invalid paths in G, where each edge $e_{ii'}$ gets assigned an action class a such that its weight $w_{ii'}(a) < w_{ii'}(a_n)$, where $a_n \neq a$ is the pseudo ground truth class for $e_{ii'}$. This constraint effectively addresses the aforementioned issue when the valid paths have significantly lower energy than the invalid paths. Alg. 3 summarizes our recursive algorithm for computing

Input: $G,\ \boldsymbol{b}_{1:N+1},\ \boldsymbol{a}_{1:N}$
Output: $\text{logadd}(\mathcal{P}^{I^c}) = \ell_T$

Initialization: $\ell_0 = 0$;
for $n = 1$ **to** N **do**
  for i' in the neighborhood of b_n **do**
    $\ell_{i'} = \infty$;
    for i in the neighborhood of b_{n-1} **do**
      for $a \in \mathcal{A}$ **do**
        temp $= \ell_i$;
        if $w_{ii'}(a) < w_{ii'}(a_n)$ **then**
          temp $= \ell_i + w_{ii'}(a)$
        end
        $\ell_{i'} = \text{logadd}(\ell_{i'}, \text{temp})$;
      end
    end
  end
end

Algorithm 3: Computing the logadd score of a subset of invalid paths $\mathcal{P}^{I_c}$, for estimating the constrained discriminative forward loss L_{CDF}.

$\text{logadd}(\mathcal{P}^{I_c})$ in (18), whereas Alg. 1 shows how to compute $\text{logadd}(\mathcal{P}^{V})$ in (18).

As L_{DF} accounts for the invalid paths, L_{CDF} further accounts for the hard invalid paths. Therefore, the model robustness is further improved by minimizing L_{CDF} which amounts to maximizing the decision margin between the valid and hard invalid paths.

5.4. Our Computational Efficiency

As summarized in Alg. 1–3, our training first runs the constrained Viterbi algorithm (see Sec. 3) to get the initial segmentation cuts with complexity $O(T^2N)$ for a video of length T and the ground-truth action sequence of length N. Then, CDFL efficiently accumulates the energy of both valid and invalid paths in G with complexity $O(\Delta^2 KN)$ for the neighborhood window size Δ and the class set size K. Therefore, our total complexity of training is $O(T^2N + \Delta^2 KN)$.

Note that prior work [22] also runs the Constrained Viterbi with complexity $O(T^2N)$, so relative to theirs our complexity is increased by $O(\Delta^2 KN)$. This additional complexity is significantly smaller than $O(T^2N)$ as $\Delta^2 K \ll T^2$. In our experimental evaluation, we get the best results for $\Delta \leq 20$ frames, whereas video length T can go to several minutes.

6. Results

Both action segmentation and alignment are evaluated on the Breakfast Actions [10], Hollywood Extended [1], and 50Salads [24] datasets. We perform the same cross-validation strategy as the state of the art, and report our average results. We call our approach CDFL, trained with loss given by (18).

Datasets. For all datasets, we use as input the pre-processed, public, unsupervised frame-level features. The same frame features are used by [8, 12, 20, 22]. The features are dense trajectories represented by PCA-projected Fisher vectors [11]. *Breakfast* [10] consists of $1,712$ videos of people making breakfast with 10 cooking activities. The cooking activities are comprised of 48 action classes. On average, every video has 6.9 action instances, and the video length ranges from a few seconds to several minutes. *Hollywood Extended* [1] contains 937 video clips from different Hollywood movies, showing 16 action classes. Each clip contains 2.5 actions on average. *50Salads* [24] has 50 very long videos showing 17 classes of human manipulative gestures. On average, each video has 20 action instances. There are $600,000$ annotated frames.

Evaluation Metrics. We use the following four standard metrics, as in [1, 7]. The mean-over-frames (**Mof**) is the average percentage of correctly labeled frames. To overcome the potential drawback that frames are dominated by background class, we compute mean-over-frames without background(**Mof-bg**) as the average percentage of correctly labeled video frames with background frames removed.

Breakfast	Mof	Mof-bg	IoU	IoD
OCDC[1]	8.9	-	-	-
CTC[8]	21.8	-	-	-
HTK [11]	25.9	-	9.8	-
ECTC [8]	27.7	-	-	-
HMM/RNN [20]	33.3	-	-	-
TCFPN [7]	38.4	38.4	24.2	40.6
NN-Viterbi [22]	43.0	-	-	-
D3TW [3]	45.7	-	-	-
Our CDFL	**50.2**	**48.0**	**33.7**	**45.4**
Hollywood Ext	Mof	Mof-bg	IoU	IoD
HTK [11]	33.0	-	8.6	-
HMM/RNN [20]	-	-	11.9	-
TCFPN [7]	28.7	34.5	12.6	18.3
D3TW [3]	33.6	-	-	-
Our CDFL	**45.0**	**40.6**	**19.5**	**25.8**
50Salads	Mof	Mof-bg	IoU	IoD
CTC[8]	11.9	-	-	-
HTK [11]	24.7	-	-	-
HMM/RNN [20]	45.5	-	-	-
NN-Viterbi [22]	49.4	-	-	-
Our CDFL	**54.7**	**49.8**	**31.5**	**40.4**

Table 1. Action segmentation evaluations on Breakfast, Hollywood Ext and 50Salads. The dash means no results reported by prior work.

Figure 3. Ground truth action sequence (take_cup, spoon_powder, pour_milk, stir_milk) (top) and our CDFL's action segmentation (bottom) on the sample test video *P03_stereo01_P03_milk* from Breakfast dataset. The background frames are marked in white. CDFL may miss the true start and end of some actions, but successfully detects the actions.

Window Size	L_{F}		L_{DF}		L_{CDF}	
	Mof	IoD	Mof	IoD	Mof	IoD
30	43.5	39.4	46.6	40.5	49.4	44.1
20	44.3	40.9	47.0	41.8	**50.2**	**45.4**
10	43.8	40.0	46.2	41.3	49.6	44.6
0	43.0	38.7	45.0	40.2	48.5	43.5

Table 2. Mof and IoD evaluations on Breakfast for different neighborhood window sizes and different losses. CDFL with neighbor-window size of 20 shows the best result.

The intersection over union (**IoU**) and the intersection over detection (**IoD**) are computed as IoU = $|GT \cap D|/|GT \cup D|$, and IoD = $|GT \cap D|/|D|$, where $|GT|$ denotes the extent of the ground truth segment and $|D|$ is the extent of a correctly detected action segment.

Training. We train a single-layer GRU with 64 hidden units in 10^5 iterations, where for each iteration one training video is randomly selected. The initial learning rate of 0.01 is decreased to 0.001 at the $60,000$th iteration. The mean action lengths λ_a in (3), and the action priors $p(a)$ in (2) are estimated from the history of pseudo ground truths. Unlike [22], we do not use the history of pseudo ground truths for computing loss in the current iteration. Consequently, our training time per iteration is less than that of [22].

6.1. Action Segmentation

Tab. 1 compares CDFL with the state of the art. From the table, CDFL achieves the best performance in terms of all the four metrics. Fig. 3 qualitatively compares the ground truth and CDFL's output on an example test video in Breakfast dataset. As can be seen, CDFL typically misses the true start or end of actions by only a few frames. In general, CDFL successfully detects most action occurrences.

Ablation Study for Action Segmentation. Tab. 2 compares our action segmentation performance on Breakfast when using different sizes of the neighborhood window placed around the initial segmentation cuts (as explained in Sec. 4) and different loss functions (as specified in Sec. 5). From the table, training by accounting for invalid paths in L_{DF} and L_{CDF} gives better performance than only accounting for valid paths in L_{F}. In addition, considering neighboring frames for action boundary refinement within a window around the initial segmentation cuts gives better performance than taking into account only a single optimal path in the segmentation graph when the window size is 0. The best test performance is achieved using L_{CDF} with window size of 20 in training.

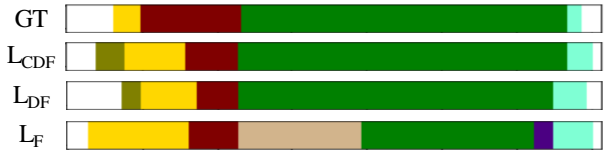

Figure 4. Top-down, the rows correspond to ground truth sequence of actions (pour_oil, crack_egg, fry_egg, put_egg2plate) and our action segmentations with neighbor-window size of 20 on the sample video *P03_cam01_P03_friedegg* from Breakfast dataset using L_{CDF}, L_{DF} and L_{F}, respectively. The background frames are marked in white. The result for L_{CDF} is the best.

window size	$\alpha = 0$	$\alpha = 0.1$	$\alpha = 0.2$	$\alpha = 0.3$
30	43.5	46.6	38.8	34.0
20	44.3	47.0	40.7	35.5
10	43.8	46.2	41.0	35.4
0	43.0	45.0	39.1	33.5

Table 3. Mof evaluations on Breakfast using L_{DF} in training with different regularization factors and neighbor-window sizes.

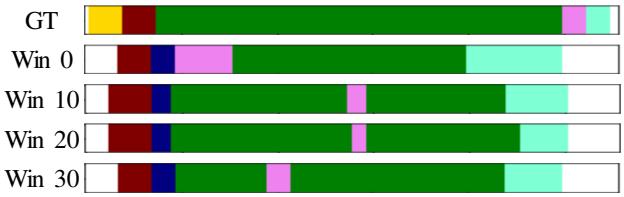

Figure 5. Ground truth action sequence (pour_oil, crack_egg, fry_egg, take_plate, put_egg2plate) (top) and CDFL's action segmentations using different neighbor-window sizes on the sample test video *P04_webcam02_P04_friedegg* from Breakfast. The background frames are marked in white. The window size of 20 gives the best performance.

Fig. 5 illustrates the CDFL's action segmentations on a sample test video from the Breakfast Action dataset using different window sizes and L_{CDF}. As can be seen, considering neighboring frames around the anchor segmentation improves performance.

Tab. 3 shows how different regularization factors α in L_{DF} affect our action segmentation on the Breakfast Action dataset, for different neighbor-window sizes. As expected, using small α in training tends to give better performance. The best accuracy is achieved with $\alpha = 0.1$ and window size of 20.

6.2. Action Alignment

Tab. 4 shows that CDFL outperforms the state-of-the-art approaches in action alignment on the three benchmark

Breakfast	Mof	Mof-bg	IoU	IoD
ECTC[8]	35.0	-	-	45.0
HTK [11]	43.9	-	26.6	42.6
OCDC [1]	-	-	-	23.4
HMM/RNN [20]	-	-	-	47.3
TCFPN [7]	53.5	51.7	35.3	52.3
D3TW [3]	57.0	-	-	56.3
Our CDFL	**63.0**	**61.4**	**45.8**	**63.9**
Hollywood Ext	Mof	Mof-bg	IoU	IoD
ECTC[8]	-	-	-	41.0
HTK [11]	49.4	-	29.1	46.9
OCDC [1]	-	-	-	43.9
HMM/RNN [20]	-	-	-	46.3
TCFPN [7]	57.4	36.1	22.3	39.6
NN-Viterbi [22]	-	-	-	48.7
D3TW [3]	59.4	-	-	50.9
Our CDFL	**64.3**	**70.8**	**40.5**	**52.9**
50Salads	Mof	Mof-bg	IoU	IoD
Our CDFL	**68.0**	**65.3**	**45.5**	**58.7**

Table 4. Action alignment evaluations on Breakfast, Hollywood Ext and 50Salads. The dash indicates that no results reported by prior work.

Figure 6. Ground truth action sequence (StandUp,SitDown, DriveCar, OpenDoor, OpenDoor, HugPerson) (top) and our action alignments (bottom) on the sample video *0261* from Hollywood Extend. The background frames are marked in white. CDFL typically achieves a good action alignment.

datasets. Fig. 6 illustrates that CDFL is good at action alignment on a sample test video from Hollywood Extended.

Ablation Study for Action Alignment. Tab. 5 presents our alignment results using different loss functions as specified in Sec. 5, and different neighbor-window sizes on Hollywood Ext. From the table, training with L_{DF} and L_{CDF} that account for invalid paths, outperforms our approach trained with L_{F}. In addition, taking into account neighboring frames around segmentation cuts of the initial segmentation (i.e., window size is greater than 0) improves performance relative to the case when window size is 0. The best performance is achieved using L_{CDF} with the window sizes of 6 in training.

Fig. 7 illustrates that CDFL gives good action alignment results on the sample test video from Hollywood Ext, using L_{CDF} and window size of 6 in training.

7. Conclusion

We have extended the existing work on weakly supervised action segmentation that uses an HMM and GRU for

Window size	L_{F}	L_{DF}	L_{CDF}
8	48.7	49.8	51.6
6	49.3	50.5	**52.9**
4	49.0	50.0	52.0
2	48.5	49.5	50.7
0	48.7	49.3	49.8

Table 5. IoD evaluations of our approach in action alignment on Hollywood Extended using different loss functions and different neighbor-window sizes in training. Using CDFL with neighbor-window size of 6 shows the best result.

Figure 7. Ground truth action sequence (OpenDoor, OpenDoor, OpenCarDoor) (top) and CDFL's action alignments on the sample test video *0361* from Hollywood Extended, when trained using varying window sizes. The background frames are marked in white. Using CDFL and neighbor-window size of 6 gives the best results.

labeling video frames by formulating a new energy-based learning on a video's segmentation graph. The graph is constructed so as to facilitate computation of loss, expressed in terms of the energy of valid and invalid paths representing candidate action segmentations. Our key contribution is the new recursive algorithm for efficiently computing the accumulated energy of exponentially many paths in the segmentation graph. Among the three loss functions that we have defined, and evaluated, the CDFL – specified to maximize the discrimination margin between valid and high-scoring invalid paths – gives the best performance. A comparison with the state of the art on both action segmentation and action alignment tasks, for the Breakfast Action, Hollywood Extended and 50Salads datasets, supports our novelty claim that using our CDFL in training gives superior results than a loss function estimated on a single inferred segmentation, as done by prior work. Our results on both action segmentation and action alignment tasks also demonstrate advantages of considering many candidate segmentations in neighbor-windows around the initial video segmentation, and maximizing the margin between all valid and hard invalid segmentations. Our small increase in complexity relative to that of related work seems justified considering our significant performance improvements.

Acknowledgement. This work was supported in part by DARPA XAI Award N66001-17-2-4029 and AFRL STTR AF18B-T002.

References

[1] Piotr Bojanowski, Rémi Lajugie, Francis Bach, Ivan Laptev, Jean Ponce, Cordelia Schmid, and Josef Sivic. Weakly supervised action labeling in videos under ordering constraints. In *European Conference on Computer Vision*, pages 628–643. Springer, 2014.

[2] Léon Bottou and Yann LeCun. Graph transformer networks for image recognition. *Bulletin of the 55th Biennial Session of the International Statistical Institute (ISI)*, 2005.

[3] Chien-Yi Chang, De-An Huang, Yanan Sui, Li Fei-Fei, and Juan Carlos Niebles. D3tw: Discriminative differentiable dynamic time warping for weakly supervised action alignment and segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3546–3555, 2019.

[4] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.

[5] Ronan Collobert. Deep learning for efficient discriminative parsing. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, pages 224–232, 2011.

[6] Li Ding and Chenliang Xu. Tricornet: A hybrid temporal convolutional and recurrent network for video action segmentation. *arXiv preprint arXiv:1705.07818*, 2017.

[7] Li Ding and Chenliang Xu. Weakly-supervised action segmentation with iterative soft boundary assignment. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6508–6516, 2018.

[8] De-An Huang, Li Fei-Fei, and Juan Carlos Niebles. Connectionist temporal modeling for weakly supervised action labeling. In *European Conference on Computer Vision*, pages 137–153. Springer, 2016.

[9] Oscar Koller, Sepehr Zargaran, and Hermann Ney. Re-sign: Re-aligned end-to-end sequence modelling with deep recurrent cnn-hmms. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4297–4305, 2017.

[10] 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*, pages 780–787, 2014.

[11] Hilde Kuehne, Juergen Gall, and Thomas Serre. An end-to-end generative framework for video segmentation and recognition. In *Applications of Computer Vision (WACV), 2016 IEEE Winter Conference on*, pages 1–8. IEEE, 2016.

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

[13] Yann Le Cun, Leon Bottou, and Yoshua Bengio. Reading checks with multilayer graph transformer networks. In *Acoustics, Speech, and Signal Processing, 1997. ICASSP-97., 1997 IEEE International Conference on*, volume 1, pages 151–154. IEEE, 1997.

[14] Colin Lea, Austin Reiter, René Vidal, and Gregory D Hager. Segmental spatiotemporal cnns for fine-grained action segmentation. In *European Conference on Computer Vision*, pages 36–52. Springer, 2016.

[15] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[16] Peng Lei and Sinisa Todorovic. Temporal deformable residual networks for action segmentation in videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6742–6751, 2018.

[17] Mengxi Lin, Nakamasa Inoue, and Koichi Shinoda. Ctc network with statistical language modeling for action sequence recognition in videos. In *Proceedings of the on Thematic Workshops of ACM Multimedia 2017*, pages 393–401. ACM, 2017.

[18] Colin Lea Michael D Flynn René and Vidal Austin Reiter Gregory D Hager. Temporal convolutional networks for action segmentation and detection. In *IEEE International Conference on Computer Vision (ICCV)*, 2017.

[19] Alexander Richard and Juergen Gall. Temporal action detection using a statistical language model. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3131–3140, 2016.

[20] 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*, 2017.

[21] 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*, 2018.

[22] Alexander Richard, Hilde Kuehne, Ahsan Iqbal, and Juergen Gall. Neuralnetwork-Viterbi: A framework for weakly supervised video learning. In *IEEE Conf. on Computer Vision and Pattern Recognition*, volume 2, 2018.

[23] Bharat Singh, Tim K Marks, Michael Jones, Oncel Tuzel, and Ming Shao. A multi-stream bi-directional recurrent neural network for fine-grained action detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1961–1970, 2016.

[24] 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. ACM, 2013.

[25] Serena Yeung, Olga Russakovsky, Greg Mori, and Li Fei-Fei. End-to-end learning of action detection from frame glimpses in videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2678–2687, 2016.