Computer Vision and Image Understanding
journal homepage: www.elsevier.com

Weakly supervised action segmentation with effective use of attention and self-attention.

Yan Bin Ng, Basura Fernando**

*Institute of High Performance Computing, A*STAR, 1 Fusionopolis Way #08-10 Connexis North, Singapore 138632*

ABSTRACT

This paper generates human action sequences using a novel hybrid sequence-to-sequence model that outputs a sequence of actions in the chronological order of the actions being performed in the longer activity of a given video. At test time, our models are able to generate action for each frame using weak supervision. We evaluate several sequence-to-sequence models to solve this task and demonstrate that they are able to solve action segment generation tasks on three challenging action recognition datasets. We present how to use self-attention and standard attention mechanisms with known sequence-to-sequence models for weakly supervised video action segmentation. Our new architecture is effective for weakly supervised action segmentation that uses a combination of recurrent and transformer-based sequence-to-sequence models. Our architecture consists of Transformers and GRU encoders to encode temporal information and we use self-attention and standard attention during the decoding process. We introduce an effective positional weight prior to further improve action segmentation performance. Using this architecture and two types of attention along with positional weight priors, we obtain state-of-the-art results on Breakfast and 50Salads datasets for weakly supervised action segmentation.

1. Introduction

Typically, human action recognition models aim to recognize a *set of predefined human actions* in a given well segmented video (Laptev, 2005; Wang and Schmid, 2013; Simonyan and Zisserman, 2014; Ji et al., 2012). To better understand human actions, many related problems have been investigated, e.g. action detection (Jiang et al., 2014; Yeung et al., 2016), spatial-temporal action localization (Tian et al., 2013), action segmentation (Lea et al., 2017), and early action prediction (Fernando and Herath, 2021; Shi et al., 2018). At some level, all these problems have to classify videos into action categories. Recently, more challenging action understanding problems have been investigated, e.g., video captioning (Venugopalan et al., 2015), text-based temporal activity localization (Wang et al., 2019), weakly supervised alignment of video and text (Bojanowski et al., 2015), and complex activity recognition (Hussein et al., 2019).

Majority of natural videos consist of action sequences and it is instinctive to describe them using action sequences rather than a set of actions. If we were asked to explain the activity shown in the video of Figure 1, it is likely that we would say "women opens fridge, takes out, opens drawer, takes out, and so on..". Here the order is important as it implicitly embodies the temporal order of actions being performed. A model that is able to generate sequences of actions would better explain the video in a way that is more natural to humans. Besides, a model's ability to generate sequences of actions from a video has many applications, e.g. robotic learning from demonstration(Argall et al., 2009), common sense knowledge generation(Goyal et al., 2017) (e.g. open fridge implies taking something before closing it) and to search videos containing a specific sequence of actions (find soccer videos of having tackle followed by red card). Specifically, in the case of robotic learning from demonstration, we may be able to generate a sequence of human actions needed to complete a task using these models just by processing the video. Afterwards, a robot might be able to generate a task plan using these instructional actions and learn from human demonstrations. We present a sequence-to-sequence model to generate action sequences and then using attention mechanisms, we perform weakly supervised action segmentation.

First, we analyse the problem where we are given only the video and ground truth action sequence at training time. During test time, the model outputs the correct sequence of actions as humans do. We call this task action sequence generation. Action sequence generation models can answer the question "what actions are needed to perform activity X?" (see also

**Corresponding author: Tel.: +65-6517-7846;
e-mail: fernando_basura@ihpc.a-star.edu.sg (Basura Fernando)

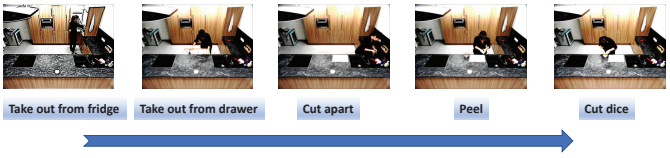

Fig. 1. Illustration of explaining a video using sequence of actions. If a method can generate sequence of actions needed to perform a task such as peeling a vegetable, then we may be able to generate a task plan for robots from those action sequences. This would allow robots to learn from demonstration.

Figure 1). Training a model that is able to output an accurate action sequence without precise temporal annotations is challenging. The model has to learn complex temporal dynamics and relationships between actions of the video to output correct action sequence. Furthermore, it has to implicitly learn when and where actions do happen (and do not), only using the input action sequences. This is a difficult learning task, as one-to-one correspondences between frames and actions are not given.

Second, at test time, our model outputs labels for each frame, even though we do not use frame level annotation at training time. This task is called *weakly supervised action segmentation* (Huang et al., 2016; Richard et al., 2017; Ding and Xu, 2018; Richard et al., 2018; Chang et al., 2019; Li et al., 2019). In a way, our model has to align input frame data with semantic action sequence while implicitly learning each human action category. Ideally, the model should learn complex relationships and inter-dependencies between actions to further improve the performance. The most challenging task is to determine the number of actions (that is the length of output action sequence) within the video. Too many or too few actions in the predicted sequence would significantly hinder the performance. As indicated in our experiments, traditional state-of-the-art action classifiers (e.g., I3D) usually tends to generate very long inaccurate action sequences. Even though weakly supervised action segmentation is more "human like" task, it is a challenging one for the machines.

As shown in a recent study, human action boundaries are ambiguous even for humans Sigurdsson et al. (2017) and therefore training and evaluation of supervised action detection becomes a challenging task. In contrast, our task only aims at generating the sequence of actions and we only penalize for the wrong order of actions ignoring action boundaries explicitly. Therefore, obtaining annotations for our task is somewhat easier, practical and potentially results in consistent and accurate annotations. A model that is trained to generate action sequences has to learn action boundaries *implicitly*, however, the notion of "action boundaries" are not used explicitly during training, somewhat similar to Weakly Supervised Action Detection (WSAD) (Paul et al., 2018; Fernando et al., 2020) and segmentation (Huang et al., 2016; Richard et al., 2017). Models that generate a sequence of actions for a video has been explored before in weakly supervised action understanding (Bojanowski et al., 2014, 2015) using textual scripts or using action sequence annotations Huang et al. (2016). Similar to (Huang et al., 2016; Richard et al., 2017) we also train models that are able to generate sequence of actions given the video. In contrast to prior work, we formulate this task as a sequence-to-sequence problem using two types of attention mechanisms, the self-attention and standard feature attention. In particular, the standard attention is applied over the encoder and decoder hidden states to identify relevant frames for each high level coarse action prediction. Self-attention is employed over the input feature sequence and the decoder hidden states by considering each decoder hidden state as the query, and input features as the key. The encoder hidden states are used as the values. These two types of attentions methods are complementary to each other. We also propose a new positional prior which allows us to align coarse action predictions with frames in fine manner using attention mechanisms. Our sequence-to-sequence model makes use of three encoders. Two of the encoders use GRU cells, one with self-attention mechanism and the other one with standard attention. The third encoder uses a Transformer model (Vaswani et al., 2017). Our model also has three GRU decoders and each decoder takes feedback from all three decoders from the previous timestep. We show that this hybrid architecture is more effective than individual sequence-to-sequence models. Specifically, we take advantage of effective properties of both recurrent encoders and modern transformers to perform weakly supervised action segmentation.

The contributions of this paper are as follows: First, we propose an effective architecture for weakly supervised action segmentation. Our Architecture consists of a hybrid of GRU and Transformer encoders and GRU decoders. Second, we show a mechanism to combine standard attention and self-attention to obtain weakly supervised action segmentation. Specifically, we propose a new self-attention-based recurrent encoder-decoder model that is complimentary to Transformers and traditional attention-based encoder-decoder models. Third, we show a new positional weight prior that allows us to improve action segmentation performance when used with attention weights. Using all these improvements and temporal connectionist loss, we obtain state-of-the-art results for weakly supervised action segmentation on Breakfast dataset. Furthermore, we evaluate the impact of sequence-to-sequence models on action segmentation task on Charades, MPII Cooking and ActivityNet 1.3 datasets.

2. Related work

We propose to tackle the action segmentation problem using sequence-to-sequence models and show effective use of attention for weakly supervised action segmentation. Several CNN and RNN based encoders have been used for solving video understanding problems (Du et al., 2015; Veeriah et al., 2015; Donahue et al., 2015; Hasan and Roy-Chowdhury, 2015). A bi-directional RNN is used for action detection in (Singh et al., 2016). Similar to other methods that use LSTMs/RNNs for action understanding tasks, this method also takes the video sequence as input and produces a sequence of predicted actions for each frame or segment. RNN model is trained with one-to-one input-output sequence correspondences. If the input video sequence has n number of elements, usually most action recognition methods that uses RNNs would output n number of action predictions and then aggregate them to make the

final action classification prediction (Singh et al., 2016; Donahue et al., 2015). Authors in (Liu et al., 2019) also make use of encoder-decoder architecture for event-detection in videos. However, they apply mean pooling over the decoder to obtain event prediction and therefore not generating a sequence of events/actions for a given video. Therefore, our work is different from theirs. Transformers are also used for various action analysis tasks such as action localization (Girdhar et al., 2019), video captioning (Zhou et al., 2018), and video retrieval (Gabeur et al., 2020). To the best of our knowledge, we are the first to use both transformers and recurrent encoders for weakly supervised action segmentation with two types of attention mechanisms. Our approach to action understanding differs from main stream action detection (Yeung et al., 2016; Singh et al., 2016), action segmentation (Shi et al., 2008) and localization methods (Hou et al., 2017) due to the nature of supervision used. The output of these methods can be further processed to align with the action-sequence, e.g. using clustering. However, these methods use precise temporal annotations during training and therefore different from our model and the task. Furthermore weakly supervised action detection methods can not generate a sequence of actions without further processing (Paul et al., 2018; Nguyen et al., 2018; Fernando et al., 2020; Wang et al., 2017), besides these methods do not make use of chronological order of actions during training. Perhaps weakly supervised action segmentation is the closest to our problem (Huang et al., 2016; Richard et al., 2017; Ding and Xu, 2018; Richard et al., 2018; Chang et al., 2019; Li et al., 2019). Weakly supervised action segmentation is challenging as it needs to infer temporal boundaries using action sequences only. Chang et al. (2019) proposed discriminative and differentiable dynamic time warping for weakly supervised action segmentation by introducing a new differentiable dynamic programming method and a new alignment loss. In contrast, we rely on the attention mechanism and positional prior with temporal connectionist loss (Graves et al., 2006) to handle the sequence alignment. Perhaps in terms of approach, methods such as discriminative and differentiable dynamic time warping is complementary to our model as well. It should be noted that Huang et al. (2016) also used a modified version of temporal connectionist Graves et al. (2006) loss for weakly supervised action segmentation. In this work we build upon these pioneering works Huang et al. (2016) with modern attention mechanisms and sequence encoding methods such as transformers. We demonstrate how to extend sequence-to-sequence models to solve weakly supervised action segmentation problem with two types of attention and obtain state-of-the-art results without any further post-processing.

3. Weakly supervised action segmentation model

3.1. Problem

Given a RGB video sequence $X = \langle x_1, x_2, \cdots, x_t, \cdot x_n \rangle$ of length n and the corresponding sequence of coarse human actions $Y = \left\langle y_1, y_2, \cdots y_p \right\rangle$ of length p, first we learn a model that generates the action sequence Y from the video sequence X. Here x_i is a RGB frame, y_j is a categorical human action, $\mathcal{Y}$ is the set of human actions and each action $y_j \in \mathcal{Y}$. The number of human action categories is C, i.e. $|\mathcal{Y}| = C$. We do not have access to the labels for individual frames during the training time, however we aim to obtain labels for individual frames at test time, and therefore our objective is to solve weakly supervised action segmentation (or localization). For short, we call a coarse human action sequence Y by *action sequence*. During training the model has to learn a set of parameter Θ such that it can predict the action sequence as follows:

$$\left\langle y_1, y_2, \cdots y_p \right\rangle = \Phi(\langle x_1, x_2, \cdots x_n \rangle, \Theta) \tag{1}$$

where both X and Y are of arbitrary length sequences. Therefore, the task in equation 1 is a sequence-to-sequence one (Sutskever et al., 2014) where the input sequence consists of three dimensional tensors (RGB frames) and the output sequence consists of categorical symbols (action classes) corresponding to action segments. Then, the problem is to infer the labels of each individual frame x_t and obtain the action score sequence for all frames denoted by $\hat{\mathbf{S}} = \langle \hat{\mathbf{s}}_\mathbf{1}, \cdots, \hat{\mathbf{s}}_\mathbf{t}, \cdots, \hat{\mathbf{s}}_\mathbf{n} \rangle$ at test time. We have access to the ground truth coarse action sequence Y and we make use of action sequence generator model $\Phi(X, \Theta)$ to produce frame level action score sequence $\hat{\mathbf{S}}$ using attention $\alpha()$, positional weight prior $\gamma()$ and Connectionist Temporal Classification (CTC) loss $CTC()$. Therefore, the overall problem can be summarised using the following model equation at high-level:

$$X \overset{\Phi(\cdot,\Theta)}{\longmapsto} \hat{Y} \overset{\alpha()\circ\gamma()\circ CTC()}{\longmapsto} \hat{\mathbf{S}} \tag{2}$$

where $\alpha() \circ \gamma() \circ CTC()$ is the composition of functions.

3.2. Attention-based solution

The objective of weakly supervised action segmentation is to obtain an action label for each frame x_i (Chang et al., 2019; Huang et al., 2016) at test time, yet these models do not use frame level annotations during training. We solve the problem of weakly supervised action segmentation using an action sequence generation model that is trained with an attention mechanism. We assume that a good action sequence generation model that is trained with attention, can solve the weakly supervised action segmentation problem. Therefore, we train an action sequence generation model as shown in equation 1 where during training, we make use of attention mechanism to align each coarse prediction y_p of equation 1 to corresponding set of frames in X using two more techniques, namely the positional weight prior and CTC loss. We use these learned attention weights to infer the action label of each frame at test time. We utilise the normalised attention weight $\alpha_{i,q}$ along with action segment score vector $\hat{\mathbf{y}}_\mathbf{q}$ (for $\{q = 1, \cdots, p\}$) to get a frame score $\hat{\mathbf{s}}_\mathbf{i}$ for the i-th frame after normalising the action segment score $\hat{\mathbf{y}}_\mathbf{q}$ by $softmax$ function as follows:

$$\hat{\mathbf{s}}_\mathbf{i} = \sum_{q=1}^{p} \gamma_{i,q} \times softmax(\hat{\mathbf{y}}_\mathbf{q}) \times \alpha_{i,q}. \tag{3}$$

where $\gamma_{i,q}$ is the **positional weight prior** obtained by following equation

$$\gamma_{i,q} = \frac{2piqn}{p^2i^2 + q^2n^2} \tag{4}$$

In the above formula, the attention weight $\alpha_{i,q}$ would tell us how much frame i is relevant for generating q-th action segment y_q. The positional weight prior ($0 \leq \gamma_{i,q} \leq 1$) allows us to better align q-th action with i-th frame by using the similarity between $(\frac{i}{n})$ and $(\frac{q}{p})$ define as follows:

$$\gamma_{i,q} = \frac{2 \times \frac{i}{n} \times \frac{q}{p}}{\left(\frac{i}{n}\right)^2 + \left(\frac{q}{p}\right)^2} \tag{5}$$

The rationale is that actions are chronologically ordered and so do the labels of the frames. The objective of positional weight prior is to explicitly enforce this prior constraint in a softer manner. Unlike Transformer positional encoding (Vaswani et al.,2062017), this constraint is enforced at action score level. The similarity measure in equation 5 is smooth, non-linear, and bounded. Therefore, this measure of similarity is effective in our context. We use this similarity as the weight prior where n is the number of frames and p is the length of generated action action sequence. In the above equation if $\frac{i}{n}$ is similar to $\frac{q}{p}$ then the influence of action symbol prediction $softmax(\hat{\mathbf{y}}_\mathbf{q})$ on the i-th frame is amplified. The normalized action symbol score ($softmax(\hat{\mathbf{y}}_\mathbf{q})$) of action at step q is propagated back to the frames in a weighted manner using frame-level attention scores and positional weight prior. We do not use frame level temporal annotations during training, yet we are able to obtain frame level action predictions during inference, hence we are performing weakly supervised action segmentation. Next we describe two types of attention mechanisms we use in our model: *attention & alignment* and *self-attention*.

3.3. Attention & alignment

During the training of model in presented in equation 1, we use the sequence encoder and a decoder with attention. For a given input feature vector sequence $\mathbf{X} = \langle \mathbf{x_1}, \mathbf{x_2}, \cdots, \mathbf{x_T} \rangle$, the encoder $f()$ produces the sequence of hidden states $\mathbf{H} = \langle \mathbf{h_1}, \mathbf{h_2}, \cdots, \mathbf{h_T} \rangle$. The hidden state at time step t is defined using the encoder function f as follows:

$$\mathbf{h_t} = f(\mathbf{x_t}, \mathbf{h_{t-1}}). \tag{6}$$

The decoder outputs an action prediction at each decoding step q denoted by $\hat{\mathbf{y}}_\mathbf{q}$. The decoder hidden state at step q is denoted by $\mathbf{h}^\mathbf{g}_\mathbf{q}$. We learn attention weights over the encoder hidden state sequence $\mathbf{H}$ for generating each decoder hidden state $\mathbf{h}^\mathbf{g}_\mathbf{q}$ and the action prediction $\hat{\mathbf{y}}_\mathbf{q}$. The weight β_i assigned to the encoder hidden state $\mathbf{h_i}$ for generating symbol $\hat{y}_q$ is obtained by the following:

$$\beta_i = tanh\left(\left[\mathbf{h_i}; \mathbf{h}^\mathbf{g}_\mathbf{q}\right]^T \times W_{att}\right) \times V \tag{7}$$

where $V \in \mathcal{R}^{1 \times D}$ and $W_{att} \in \mathcal{R}^{2D \times D}$ are learnable parameters. The notation $[\mathbf{h_i}; \mathbf{h}^\mathbf{g}_\mathbf{q}]$ is used for column vector concatenation. Thereafter, to obtain the attention weight for encoder hidden state $\mathbf{h_i}$ for generating action symbol y_q, we use softmax function over all weights $\{\beta_1, \beta_2, \cdots, \beta_T\}$ as follows:

$$\alpha_{i,q} = att\left(\mathbf{h_i}, \mathbf{h}^\mathbf{g}_\mathbf{q}\right) = \frac{exp(\beta_i)}{\sum_{j=1}^{T} exp(\beta_j)}. \tag{8}$$

As only a handful of hidden states in sequence $\mathbf{H}$ contributes to generate the output symbol y_q, it makes sense to use attention over $\mathbf{H}$ when generating the next symbol using the decoder $g()$. To do that, we propose to compute a context vector which is a weighted sum of encoder hidden states where the weight is given by equation 8. For generating q^{th} action symbol, then the context vector $\mathbf{c}^\mathbf{g}_\mathbf{q-1}$ is obtained by equation 9.

$$\mathbf{c}^\mathbf{g}_\mathbf{q-1} = \sum_{j=1}^{T} \alpha_{i,q-1} \mathbf{h_j}. \tag{9}$$

After that, the decoder $g()$ takes the context and action sequence vectors $\hat{\mathbf{y}}_\mathbf{q-1}$ as follows:

$$\mathbf{h}^\mathbf{g}_\mathbf{q} = g\left(\left[\hat{\mathbf{y}}_\mathbf{q-1}; \mathbf{c}^\mathbf{g}_\mathbf{q-1}\right], \mathbf{h}^\mathbf{g}_\mathbf{q-1}\right) \tag{10}$$

where $[a; b]$ denotes vector concatenation. As shown in eq. 11, we use linear mapping (U) over three concatenated vectors, i.e., the hidden state of the decoder, the attention weighted context vector and the previous action vector representation to generate next action symbol $\hat{y}_q$.

$$\begin{aligned} \hat{\mathbf{y}}_\mathbf{q} &= U\left[\mathbf{h}^\mathbf{g}_\mathbf{q}; \mathbf{c}^\mathbf{g}_\mathbf{q-1}; \hat{\mathbf{y}}_\mathbf{q-1}\right] \\ \hat{y}_q &= \texttt{argmax}\ \hat{\mathbf{y}}_\mathbf{q} \end{aligned} \tag{11}$$

This model that uses GRU Encoder-Decoder with attention and alignment is called **GRU-EDAA**.

3.4. Self attention

Self attention applies the attention mechanism at the input. For a given input feature $\mathbf{x_t}$, the attention is computed using input sequence $\mathbf{X} = \langle \mathbf{x_1}, \mathbf{x_2}, \cdots \mathbf{x_n} \rangle$ and the encoder hidden sequence $\mathbf{H} = \langle \mathbf{h_1}, \mathbf{h_2}, \cdots \mathbf{h_n} \rangle$. We use the decoder (the hidden state denoted by $\mathbf{h}^\mathbf{g}_\mathbf{q-1}$) to generate the action sequence symbol y_q at decoding step q. Then the context vector $\mathbf{c}^\mathbf{g}_\mathbf{q-1}$ for generating the next action symbol y_q is obtained as follows using self attention:

$$\mathbf{c}^\mathbf{g}_\mathbf{q-1} = softmax\left(\frac{\mathbf{h}^\mathbf{g}_\mathbf{q-1}\mathbf{X}^T}{\sqrt{D}}\right)\mathbf{H} \tag{12}$$

where $\mathbf{X}$ and $\mathbf{H}$ are now denoted as matrices (with slight abuse of notation) and D is the dimension of hidden states and the input feature. Note that the self attention weight of the i-th frame for generating q-th action symbol y_q is obtained by the i-th element of score vector $\alpha_\mathbf{q}$ of the followsing:

$$\alpha_q = softmax\left(\frac{\mathbf{h}^\mathbf{g}_\mathbf{q-1}\mathbf{X}^T}{\sqrt{D}}\right) \tag{13}$$

The decoder $g()$ generates the next hidden state using equation 10. The rest of the model follows equations 11 to generate action symbol y_q. We call this self-attention-based GRU encoder-decoder model by **SAGD**.

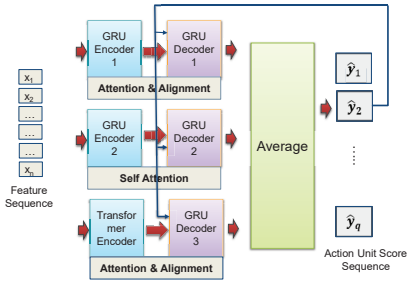

Fig. 2. Visual illustration of our weakly supervised action segmentation model with attention.

3.5. Weakly supervised action segmentation model

Weakly supervised action segmentation is a challenging task and therefore we make use of a hybrid architecture of encoder-decoder models as illustrated in figure 2. Our model consists of two GRU encoder-decoders with two attention mechanisms, alignment and attention (GRU-EDAA) and self-attention (SAGD) which are explained in section 3.3 and 3.4 respectively. We also use a Transformer encoder (Vaswani et al., 2017) and a GRU decoder with alignment and attention.

In the Transformer model, each input sequence is processed by three weight matrices; the query weights W_{QT}, the key weights W_{KT}, and the value weights W_{VT}. Each vector element in the sequence is multiplied with each of the three weight matrices to produce a query vector, a key vector, and a value vector. Then the Transformer model uses scalar dot-product with softmax normalisation similar to equation 11 to obtain the attention weights. The output of the Transformer is a weighted combination of inputs where the weights are obtained by attention. One set of $\{W_Q^T, W_K^T, W_V^T\}$ weight matrices is called an attention head, and each layer in a Transformer there can be multiple attention heads.

We exploit advantages of all three state-of-the-art sequence-to-sequence models to generate action segmentation in a weakly supervised manner. While GRU-EDAA and SAGD components are of recurrent nature, the Transformer encoder exploits the associations between input data sequences therefore both models are complementary to each other. The full model has three encoders and three GRU decoders. All three encoders take the input sequence and generate three encoder hidden sequences. Each decoder takes corresponding encoder hidden sequence and generates the action symbol at each step q. The final action symbol score for y_q is the average of all three scores. Note that this final score vector is also fed to the GRU decoders as input which makes sure that each decoder gets feedback from all other decoders using equation 10 in the decoding step. At test time, we obtain three action prediction scores for each frame using equation 3, one for each model. Then we take the average of them as the final frame prediction for action segmentation task. The way we obtain the attention score for GRU Encoder-Decoder with alignment and attention (GRU-EDAA) is given by equation 8. For self-attention, the weight $\alpha_{i,q}$ is obtained by equation 13 similar to SAGD model.

3.6. Loss function

Let us denote the sequence of action scores obtained for each frame using equation 3 by $\hat{\mathbf{S}} = \langle \hat{\mathbf{s}}_\mathbf{1}, \cdots, \hat{\mathbf{s}}_\mathbf{n} \rangle$. To further improve the alignment of actions frame score sequence $\hat{\mathbf{S}}$ and the ground truth action symbol sequence $Y = \langle y_1, y_2, \cdots y_p \rangle$, we use the Connectionist Temporal Classification Loss (CTC) (Graves et al., 2006). Note that the length of the score sequence $\hat{\mathbf{S}}$ is n and the length of the target label sequence Y is p. CTC loss provides a natural way to align these two different length sequences and calculate a loss. CTC calculates the loss by summing over the probability of each alignment of input sequence to the target sequence. This way, it produces a differentiable loss with respect to the input. There is "many-to-one" alignment of input to the target which constrains the length of the target sequence to be smaller than the input length. Let us denote the set of plausible alignments between the input $\hat{\mathbf{S}}$ and the target sequence Y by $\mathcal{A}$. CTC estimates the conditional probability $P(Y|\hat{\mathbf{S}})$ by marginalising over all possible alignments $\mathcal{A}$

$$P(Y|\hat{\mathbf{S}}) = \sum_{A \in \mathcal{A}} \prod_i (\hat{\mathbf{s}}_\mathbf{i} \cdot \mathbf{a}_\mathbf{i}^\mathbf{T}) \tag{14}$$

where $\mathbf{a_t}$ is the one hot vector action representation of the alignment $A = \langle \mathbf{a_1}, \cdots, \mathbf{a_n} \rangle$ from frames to the target action sequence. Some alignments are highly probable while some others are highly improbable. CTC loss encourages the model to maximise the likelihood of probable alignments and therefore minimise the negative log likelihood as follows:

$$CTC\left(\hat{\mathbf{S}}, \mathbf{Y}\right) = -logP(Y|\hat{\mathbf{S}}). \tag{15}$$

The CTC algorithm enumerates all plausible alignments from target Y to compute the loss. CTC algorithm makes the loss estimation feasible by using a dynamic programming technique (Graves et al., 2006). We make use of both cross entropy loss (CE) and CTC loss in our model as follows:

$$Loss = \sum_{q=1}^{p} CE\left(\hat{\mathbf{y}}_\mathbf{q}, \mathbf{y_q}\right) + \beta \times CTC\left(\hat{\mathbf{S}}, \mathbf{Y}\right) \tag{16}$$

where β is a regularisation hyper-parameter and set to 0.1. Note that the CTC loss can align the frame scores sequence $\hat{\mathbf{S}}$ with the ground truth action label sequence $\mathbf{Y}$. Therefore, in this loss we can utilise the frame scores. However, when we utilise cross-entropy loss, we can use only action symbol predictions to compute it as we don't know the alignment during training. Our model is trained with start-of-sequence and end-of-sequence tokens during training. Therefore, unlike most prior methods, our models do not need explicit mechanisms to find the length of the action sequence.

4. Experiments

4.1. Baseline comparisons

Our first goal is to find good sequence-to-sequence models to solve the problem denoted in equation 1 to solve coarse action segmentation. It should be noted that coarse action segmentation and action sequence generation are similar tasks. We

evaluate coarse action segmentation performance of sequence-to-sequence models using three datasets, Charades (Sigurdsson et al., 2016), MPII Cooking (Rohrbach et al., 2012) and ActivityNet 1.3 (Fabian Caba Heilbron and Niebles, 2015). Here our assumption is that a good action sequence generation model (a model that can generate coarse action segmentation) can solve weakly supervised action segmentation tasks effectively using attention, positional weight prior and CTC loss. Therefore, first we find a good sequence-to-sequence model to generate action sequences. The ground truth action sequence is obtained using the start time of each action in the given video. The average action sequence length (number of action segments) is 6.8, 46.0 and 2.0 actions per video respectively on these three datasets. We use unit accuracy in segmentation literature (Chang et al., 2019; Huang et al., 2016) to evaluate models. We use pre-trained I3D network (Carreira and Zisserman, 2017) features trained on Kinetics dataset (Kay et al., 2017). The hidden size of all models are set to 512. We train our models with <SOS>, <EOS> and padding symbols. Our action sequence generation models are trained with batch size of 32 videos using the Adam optimizer with a learning rate of $1e^{-3}$ and early stopping on a validation set. We train several baseline sequence-to-sequence models using cross entropy loss only. We also train a fully supervised I3D (Carreira and Zisserman, 2017) model which makes use of frame level annotations to predict action for each 32 frame clips as a baseline. In contrast to our sequence-to-sequence models, this fully supervised I3D model uses frame level action annotations during training. Afterwards, we post-process the prediction score sequence to obtain an action sequence using connected component algorithm where two adjacent video clips are joined if they share the same action for action segmentation. We also train a fully supervised ResNet(2+1)D (Tran et al., 2018) model similar to the I3D. We use the same post processing sequence summary method for ResNet(2+1)D. Third, we use the mean pooled I3D features as input and use a LSTM model (`LSTM-Mean`) to output the sequence of action symbols. We compare several baseline models as shown in Table 1. Transformer works well in two datasets while GRU-EDAA performs better in Charades dataset. LSTM-ED does not work that well. LSTM-Mean performs poorly and therefore demonstrates the importance of encoding temporal evolution of video feature sequences . Surprisingly, even if the GRU-EDAA is relatively simple, it works well across all datasets. GRU-EDAA captures the temporal relationships of video features better than LSTM-ED. GRU-EDAA seems to benefit from additional 1D convolution layer in two datasets. This also suggests that these models might benefit from additional temporal modeling. Transformer model is competitive in all three datasets, yet there is no convincing winner for action segmentation task across all three datasets. All models are trained with coarse action sequence annotations except for I3D and ResNet(2+1)D models. The reason for inferior results of I3D and ResNet(2+1)D is because they generate temporally inconsistent (non-smooth), fragmented and shorter action segments. In contrast, sequence-to-sequence models learn the length of each target sequence fairly accurately and are able to capture longer temporal relationships between actions. As discussed in the introduction, to solve this task, these models need to learn complex temporal dependencies between actions and boundaries of each action in a weakly supervised manner. Methods such as I3D Carreira and Zisserman (2017) and ResNet(2+1)D Tran et al. (2018) are not trained to model these temporal dependencies as in other sequence-to-sequence models. We conclude that Transformer and GRU-EDAA are more suitable to solve action segmentation across all three datasets and therefore we propose to make use of GRU-EDAA and Transformer with both self-attention and standard attention.

Table 1. Comparison of results for coarse action segmentation on three action recognition datasets. Only unit accuracy is used to evaluate the model performance of several baseline sequence-to-sequence models.

Model	Unit Acc. (%)
Charades dataset	
I3D Carreira and Zisserman (2017) (Fully supervised)	2.55
ResNet(2+1)D Tran et al. (2018) (Fully supervised)	1.28
LSTM-Mean	4.71
(LSTM-ED) LSTM Encode-Decoder	4.48
(GRU-EDAA) GRU Encode-Decoder with Attention & Alignment	5.35
(GRU-EDAA) + 1D Conv	**5.84**
(SAGD) Self-Attention and GRU Decoder	5.50
Transformer (4 heads, 1 Encoder layer, 1 Decoder layer)	5.46
Transformer (8 heads, 3 Encoder layers, 3 Decoder layers)	4.39
ActivityNet 1.3	
I3D Carreira and Zisserman (2017) (Fully supervised)	0.02
LSTM-Mean	37.51
GRU-EDAA	45.00
GRU-EDAA + 1D Conv	48.37
SAGD	48.21
Transformer (4 heads, 1 Encoder layer, 1 Decoder layer)	51.78
Transformer (8 heads, 3 Encoder layers, 3 Decoder layers)	**52.65**
MPII Cooking	
LSTM-Mean	11.17
GRU-EDAA	14.86
GRU-EDAA + 1D Conv	13.77
Transformer (4 heads, 1 Encoder layer, 1 Decoder layer)	15.85
Transformer (8 heads, 3 Encoder layers, 3 Decoder layers)	**15.96**

4.2. Weakly supervised action segmentation

In this section we evaluate the performance of our model (Figure 2) on weakly supervised action segmentation task using Breakfast (Kuehne et al., 2014) and 50Salad datasets(Stein and McKenna, 2013). For a fair comparison, we use the precomputed features and data split provided by (Kuehne et al., 2014) and follow the protocol used in (Chang et al., 2019; Huang et al., 2016). As the video segments are very long, We also generate clusters of 20 frames following prior work. We report both frame accuracy and the unit accuracy as in prior methods. We use Adam optimiser with batch size of 32 videos. Our models are trained with start and end of sequence tokens. GRU models use hidden size of 64 in both encoder and the decoder. Transformer model also uses hidden size of 64 and uses 8 attention heads. We perform some ablation study in Table 2. We use Temporal Connectionist Loss (CTC) and the positional prior $\gamma_{i,q}$ to train all models (GRU-EDAA, SAGD, Transformer, our full model) by default. We evaluate the impact of positional prior $\gamma_{i,q}$ and the CTC loss on our model.

We see that our model obtains better results than individual models (GRU-EDAA, SAGD, Transformer). The complementary nature of the models are due to the way the sequence information is encoded. Current approaches mostly use either recur-

Table 2. Weakly supervised action segmentation on Breakfast and 50Salads dataset. Ablation study on the impact of each component of our model is shown. AA stands for "Attention & Alignment".

Method	Frame (%)		Unit (%)	
Dataset	Breakfast	50Salads	Breakfast	50Salads
GRU-EDAA (AA)	41.8	43.4	48.2	54.2
SAGD Self-Attention and GRU Decorder	39.2	41.5	47.8	53.9
Transformer and GRU Decorder (AA)	41.8	44.7	46.5	59.7
Our model	52.2	56.3	53.4	64.8
Our without positional prior and without CTC	44.4	47.2	51.6	62.2
Our with positional prior and without CTC	46.9	52.1	51.6	62.2

Table 3. Comparison with state-of-the-art for weakly supervised action segmentation on Breakfast dataset and 50Salad datasets.

Dataset	Breakfast		50Salad
Method	Frame (%)	Unit (%)	Frame (%)
CTC Huang et al. (2016)	21.8	–	11.9
ECTC Huang et al. (2016)	27.7	35.6	–
GRU reest. Richard et al. (2017)	33.3	–	45.5
TCFPN Ding and Xu (2018)	38.4	–	–
NN-Viterbi Richard et al. (2018)	43.0	–	49.4
D^3TW Chang et al. (2019)	45.7	47.4	–
CDFL Li et al. (2019)	50.2	–	54.7
Our model	**52.2**	**53.4**	**56.3**

rent models or Transformer models. However, the way in which these models capture temporal information is vastly different. Transformer models solely rely on self-attention while recurrent models use hidden states to propagate information over time. We believe both information are useful in some applications. Our hybrid model obtains massive improvements in action classes such as butter-pan (+22.9), pour-milk (+17.9), spoon-flour(+17.7) over other best individual model.

The improvement obtained by the positional weight prior is 2.5% and 4.9% respectively on Breakfast and 50Salad datasets– please also see the impact of positional weight prior shown in section 4.4 and Figure 3. The CTC loss along with our model obtain an improvement of 5.3% and 4.2% on Breakfast and 50Salad datasets respectively. It should be noted that CTC loss brings a considerable improvement, however it is highly dependant on cross-entropy loss–see equation 16. Without the cross-entropy loss in our case, the model do not perform that well. We conclude that our new architecture, the positional weight prior and the CTC loss improve results of weakly supervised action segmentation by a large margin over the baselines.

4.3. Comparison with state-of-the-art

In this section we evaluate the impact of our model presented in section 3.5 for weakly supervised video action segmentation on Breakfast and 50Salads datasets. For a fair comparison, we use the pre-computed features and data splits provided by original dataset papers and follow the protocol used in (Chang et al., 2019; Huang et al., 2016). Results are compared with state-of-the-art methods in Table 3 indicating the advantage of our model. For 50Salad dataset, we report only the frame accuracy as this is the standard practice. Unit accuracy is shown in Table 2. Our method is able to obtain state-of-the-art results in both frame accuracy and unit accuracy for both Breakfast and 50Salads datasets. Our method outperforms the prior state-of-the-art by 2.0 % and 1.6 % on Breakfast and 50Salads respectively. Therefore, our model is effective in weakly supervised action segmentation on both Breakfast and 50Salad datasets.

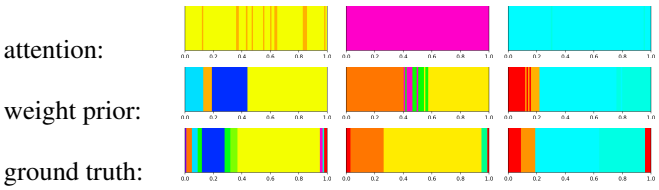

Fig. 3. The impact of weight prior on weakly supervised action segmentation on Breakfast dataset is shown. Top row shows the predictions obtained from the attention scores and in the middle row, after positional weight prior. The bottom row shows the ground truth.

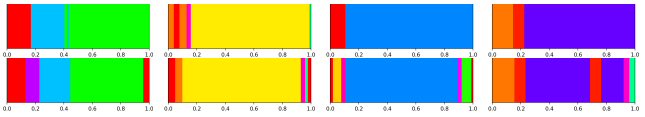

Fig. 4. Some model predictions are shown in the top row and the corresponding ground-truth is shown in the second row. The most common mistake of our model occurs when there is a longer action in the video. Then the attention scores tend to be bias towards those longer actions. Positional weight prior can correct some of them.

4.4. Qualitative analysis.

In this section we perform some qualitative analysis to show the impact of positional weight prior. We visually inspect the obtained action class for each frame of the video with and without positional weight prior. Results are shown in Figure 3. In the top row, the results without positional weight prior (i.e. only with attention) is shown and the improvements obtained after positional weight prior is shown in the middle row. The ground truth is shown in the bottom row. We show some selected examples where the positional weight prior helped to overcome the errors of attention scores by a significant margin. It should be noted that we were able to improve the results of attention scores 83% of the time when we use positional weight prior. Most interestingly, the main limitation of the attention mechanism is that it fails when there is a longer action (longer time-span) within a video. Some of these common mistakes are shown in Figure 4. Positional weight prior can correct these, however fails sometimes as shown in Figure 4.

5. Discussion and Conclusion

We investigated several sequence-to-sequence solutions for video action segment generation to output a sequence of actions for a given video in the chronological order. We obtained encouraging results on three difficult action recognition datasets, the MPII Cooking, Charades and ActivityNet datasets. Transformers and GRU Encoder-Decoder with alignment and attention method performs reasonably well on action segmentation generation problem. Then we show that with the help of attention, we can align these action sequence scores to frame scores. One of the significant aspects of our approach is that it does not require precise temporal annotations, yet it can be used to obtain frame level action predictions and therefore can be used to solve weakly supervised action segmentation. Especially, we show that use of attention on multiple feature instances: i. self-attentions between the

input feature and decoder hidden states; ii. standard attention between encoder and decoder hidden states; and then we show the complementary nature of standard-attention, self-attention and Transformers can be used for weakly supervised action segmentation tasks. We show how to strengthen standard sequence-to-sequence models using two additional attention mechanisms. We obtained improvements over prior state-of-the art on action segmentation on Breakfast and 50Salads datasets. We conclude that our action segmentation technique with positional prior, two types of attention, CTC loss and the architecture is effective.

Acknowledgment: This research/project is supported by the National Research Foundation, Singapore under its AI Singapore Programme (AISG Award No: AISG-RP-2019-010).

References

Argall, B.D., Chernova, S., Veloso, M., Browning, B., 2009. A survey of robot learning from demonstration. Robotics and autonomous systems 57, 469–483.

Bojanowski, P., Lajugie, R., Bach, F., Laptev, I., Ponce, J., Schmid, C., Sivic, J., 2014. Weakly supervised action labeling in videos under ordering constraints, in: ECCV, Springer. pp. 628–643.

Bojanowski, P., Lajugie, R., Grave, E., Bach, F., Laptev, I., Ponce, J., Schmid, C., 2015. Weakly-supervised alignment of video with text, in: ICCV, pp. 4462–4470.

Carreira, J., Zisserman, A., 2017. Quo vadis, action recognition? a new model and the kinetics dataset, in: CVPR, pp. 6299–6308.

Chang, C.Y., Huang, D.A., Sui, Y., Fei-Fei, L., Niebles, J.C., 2019. D3tw: Discriminative differentiable dynamic time warping for weakly supervised action alignment and segmentation, in: CVPR, pp. 3546–3555.

Ding, L., Xu, C., 2018. Weakly-supervised action segmentation with iterative soft boundary assignment, in: CVPR, pp. 6508–6516.

Donahue, J., Anne Hendricks, L., Guadarrama, S., Rohrbach, M., Venugopalan, S., Saenko, K., Darrell, T., 2015. Long-term recurrent convolutional networks for visual recognition and description, in: CVPR, pp. 2625–2634.

Du, Y., Wang, W., Wang, L., 2015. Hierarchical recurrent neural network for skeleton based action recognition, in: CVPR, pp. 1110–1118.

Fabian Caba Heilbron, Victor Escorcia, B.G., Niebles, J.C., 2015. Activitynet: A large-scale video benchmark for human activity understanding, in: CVPR, pp. 961–970.

Fernando, B., Herath, S., 2021. Anticipating human actions by correlating past with the future with jaccard similarity measures, in: CVPR, pp. 13224–13233.

Fernando, B., Tan, C., Bilen, H., 2020. Weakly supervised gaussian networks for action detection, in: WACV, pp. 537–546.

Gabeur, V., Sun, C., Alahari, K., Schmid, C., 2020. Multi-modal transformer for video retrieval, in: ECCV, Springer. pp. 1–14.

Girdhar, R., Carreira, J., Doersch, C., Zisserman, A., 2019. Video action transformer network, in: CVPR, pp. 244–253.

Goyal, R., Kahou, S.E., Michalski, V., Materzynska, J., Westphal, S., Kim, H., Haenel, V., Fruend, I., Yianilos, P., Mueller-Freitag, M., et al., 2017. The" something something" video database for learning and evaluating visual common sense., in: ICCV, p. 3.

Graves, A., Fernández, S., Gomez, F., Schmidhuber, J., 2006. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks, in: ICML, pp. 369–376.

Hasan, M., Roy-Chowdhury, A.K., 2015. Context aware active learning of activity recognition models, in: ICCV, pp. 4543–4551.

Hou, R., Sukthankar, R., Shah, M., 2017. Real-time temporal action localization in untrimmed videos by sub-action discovery., in: BMVC, p. 7.

Huang, D.A., Fei-Fei, L., Niebles, J.C., 2016. Connectionist temporal modeling for weakly supervised action labeling, in: ECCV, Springer. pp. 137–153.

Hussein, N., Gavves, E., Smeulders, A.W., 2019. Timeception for complex action recognition, in: CVPR, pp. 254–263.

Ji, S., Xu, W., Yang, M., Yu, K., 2012. 3d convolutional neural networks for human action recognition. IEEE TPAMI 35, 221–231.

Jiang, Y.G., Liu, J., Roshan Zamir, A., Toderici, G., Laptev, I., Shah, M., Sukthankar, R., 2014. THUMOS challenge: Action recognition with a large number of classes. `http://crcv.ucf.edu/THUMOS14/`.

Kay, W., Carreira, J., Simonyan, K., Zhang, B., Hillier, C., Vijayanarasimhan, S., Viola, F., Green, T., Back, T., Natsev, P., et al., 2017. The kinetics human action video dataset. arXiv preprint arXiv:1705.06950 .

Kuehne, H., Arslan, A., Serre, T., 2014. The language of actions: Recovering the syntax and semantics of goal-directed human activities, in: CVPR, pp. 780–787.

Laptev, I., 2005. On space-time interest points. IJCV 64, 107–123.

Lea, C., Flynn, M.D., Vidal, R., Reiter, A., Hager, G.D., 2017. Temporal convolutional networks for action segmentation and detection, in: CVPR, pp. 156–165.

Li, J., Lei, P., Todorovic, S., 2019. Weakly supervised energy-based learning for action segmentation, in: ICCV, pp. 6243–6251.

Liu, A.A., Shao, Z., Wong, Y., Li, J., Su, Y.T., Kankanhalli, M., 2019. Lstm-based multi-label video event detection. Multimedia Tools and Applications 78, 677–695.

Nguyen, P., Liu, T., Prasad, G., Han, B., 2018. Weakly supervised action localization by sparse temporal pooling network, in: CVPR, pp. 6752–6761.

Paul, S., Roy, S., Roy-Chowdhury, A.K., 2018. W-talc: Weakly-supervised temporal activity localization and classification, in: ECCV, pp. 563–579.

Richard, A., Kuehne, H., Gall, J., 2017. Weakly supervised action learning with rnn based fine-to-coarse modeling, in: CVPR, pp. 754–763.

Richard, A., Kuehne, H., Iqbal, A., Gall, J., 2018. Neuralnetwork-viterbi: A framework for weakly supervised video learning, in: CVPR, pp. 7386–7395.

Rohrbach, M., Amin, S., Andriluka, M., Schiele, B., 2012. A database for fine grained activity detection of cooking activities, in: CVPR, pp. 1194–1201.

Shi, Q., Wang, L., Cheng, L., Smola, A., 2008. Discriminative human action segmentation and recognition using semi-markov model, in: CVPR, pp. 1–8.

Shi, Y., Fernando, B., Hartley, R., 2018. Action anticipation with rbf kernelized feature mapping rnn, in: ECCV, pp. 301–317.

Sigurdsson, G.A., Russakovsky, O., Gupta, A., 2017. What actions are needed for understanding human actions in videos?, in: ICCV, pp. 2137–2146.

Sigurdsson, G.A., Varol, G., Wang, X., Farhadi, A., Laptev, I., Gupta, A., 2016. Hollywood in homes: Crowdsourcing data collection for activity understanding, in: ECCV, pp. 510–526.

Simonyan, K., Zisserman, A., 2014. Two-stream convolutional networks for action recognition in videos, in: NIPS, pp. 568–576.

Singh, B., Marks, T.K., Jones, M., Tuzel, O., Shao, M., 2016. A multi-stream bi-directional recurrent neural network for fine-grained action detection, in: CVPR, pp. 1961–1970.

Stein, S., McKenna, S.J., 2013. 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, pp. 729–738.

Sutskever, I., Vinyals, O., Le, Q.V., 2014. Sequence to sequence learning with neural networks, in: NIPS, pp. 3104–3112.

Tian, Y., Sukthankar, R., Shah, M., 2013. Spatiotemporal deformable part models for action detection, in: CVPR, pp. 2642–2649.

Tran, D., Wang, H., Torresani, L., Ray, J., LeCun, Y., Paluri, M., 2018. A closer look at spatiotemporal convolutions for action recognition, in: CVPR, pp. 6450–6459.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I., 2017. Attention is all you need, in: NIPS, pp. 5998–6008.

Veeriah, V., Zhuang, N., Qi, G.J., 2015. Differential recurrent neural networks for action recognition, in: ICCV, pp. 4041–4049.

Venugopalan, S., Rohrbach, M., Donahue, J., Mooney, R., Darrell, T., Saenko, K., 2015. Sequence to sequence-video to text, in: ICCV, pp. 4534–4542.

Wang, H., Schmid, C., 2013. Action recognition with improved trajectories, in: ICCV, pp. 3551–3558.

Wang, L., Xiong, Y., Lin, D., Van Gool, L., 2017. Untrimmednets for weakly supervised action recognition and detection, in: CVPR, pp. 4325–4334.

Wang, W., Huang, Y., Wang, L., 2019. Language-driven temporal activity localization: A semantic matching reinforcement learning model, in: CVPR, pp. 334–343.

Yeung, S., Russakovsky, O., Mori, G., Fei-Fei, L., 2016. End-to-end learning of action detection from frame glimpses in videos, in: CVPR, pp. 4507–4515.

Zhou, L., Zhou, Y., Corso, J.J., Socher, R., Xiong, C., 2018. End-to-end dense video captioning with masked transformer, in: CVPR, pp. 8739–8748.