Forecasting future action sequences with attention: a new approach to weakly supervised action forecasting

Yan Bin Ng, Basura Fernando

***Abstract*—Future human action forecasting from partial observations of activities is an important problem in many practical applications such as assistive robotics, video surveillance and security. We present a method to forecast actions for the unseen future of the video using a neural machine translation technique that uses encoder-decoder architecture. The input to this model is the observed RGB video, and the objective is to forecast the correct future symbolic action sequence. Unlike prior methods that make action predictions for some unseen percentage of video one for each frame, we predict the complete action sequence that is required to accomplish the activity. We coin this task action sequence forecasting. To cater for two types of uncertainty in the future predictions, we propose a novel loss function. We show a combination of optimal transport and future uncertainty losses help to improve results. We evaluate our model in three challenging video datasets (Charades, MPII cooking and Breakfast).**

We extend our action sequence forecasting model to perform weakly supervised action forecasting on two challenging datasets, the Breakfast and the 50Salads. Specifically, we propose a model to predict actions of future unseen frames without using frame level annotations during training. Using Fisher vector features, our supervised model outperforms the state-of-the-art action forecasting model by 0.83% and 7.09% on the Breakfast and the 50Salads datasets respectively. Our weakly supervised model is only 0.6% behind the most recent state-of-the-art supervised model and obtains comparable results to other published fully supervised methods, and sometimes even outperforms them on the Breakfast dataset. Most interestingly, our weakly supervised model outperforms prior models by 1.04% leveraging on proposed weakly supervised architecture, and effective use of attention mechanism and loss functions.

***Index Terms*—action forecasting, weakly supervised learning, action sequence forecasting**

I. Introduction.

We humans forecast others' actions by anticipating their behavior. For example by looking at the video sequence in Fig.1, we can say "the person is going towards the fridge, then probably he will open the refrigerator and take something from it". Our ability to forecast comes naturally to us. We hypothesize that humans analyze visual information to predict plausible future actions, also known as mental time travel [1]. One theory suggests that humans' success in evolution is due to the ability to anticipate the future [1]. Perhaps, we correlate prior experiences and examples with the current scenario to perform mental time travel.

Institute of High Performance Computing, A*STAR, 1 Fusionopolis Way, 16-16, Connexis North Tower, Singapore 138632 e-mail: (see https://basurafernando.github.io/).

Manuscript received April 16, 2020; revised July 16, 2020.

Fig. 1. Someone is going towards the fridge. What are the plausible future sequence of actions? Our action sequence forecasting model predicts the future action sequence { open fridge $\succ$ take something $\succ$ close fridge }, after processing the partial video. Our weakly supervised model predicts a label for each future frame without using any frame level annotations during training.

Recently, the human action prediction/forecasting problem has been extensively studied in the Computer Vision and AI community. The literature on this prediction topic can be categorized as early action [2], activity [3], [4], and event prediction [5]. In early human action prediction, methods observe an ongoing human action and aim to predict *the action in progress* as soon as possible [6] before it finishes. This problem is also known as action anticipation in the literature [7]. As these methods predict an ongoing action before it finishes, they are useful for applications when future planning is not a major requirement. In contrast, activity prediction aims at forecasting future action as soon as possible (not necessarily in the temporal order) and is useful in many robotic applications, e.g., human robot interaction. These methods can facilitate information for some level of future planning [8]. In activity prediction, some methods observe $p\%$ of the activity and then predict actions for $q\%$ of the future frames in the video. Most interestingly, these methods predict actions per-frame which limits their practical application in many cases [3]. The limitation of these methods are two fold. First, these methods need precise temporal annotations for each future frame during training. Even though this is feasible with small scale datasets, in practical applications obtaining labels for each frame is a challenging task. It is more feasible to obtain the sequence of actions without temporal extents. As an example, in Fig.1, it is easy to obtain action sequence $<$ `open, take, close` $>$ rather than precise frame level annotations. In this paper we use only those coarse action sequence labels for training. Secondly, most prior methods make the assumption about length of the video implicitly or explicitly [3], [9]. In contrast, our formulation does not make these rigid assumptions.

Alternatively, some methods observe k number of actions in an activity and then predict only the next future action [10]. However, we humans are able to forecast the future series of actions which allows us to plan for the future, (e.g. if

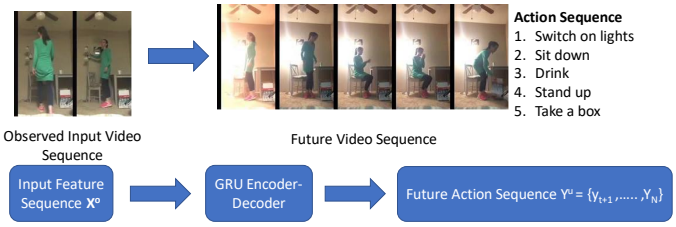

Fig. 2. A high level illustration of our action sequence forecasting solution. Given an input video, we train a GRU-based sequence-to-sequence machine translation model to forecast future action sequence. Specifically, our method should know when to stop generating actions for the future. In other words, we solve the problem of what steps (actions) are needed to finish the current activity of the person. During training we use the action sequences without temporal extents of each action. At test time, our models are able to forecast future action sequence and a label for each future unseen frame.

someone is going to cook a simple potato dish, probably we will see a sequence of actions such as peel $\succ$ cut $\succ$ wash $\succ$ boil). We humans are able to predict the future irrespective of video length or the number of frames. We aim to solve this challenging problem of forecasting future sequence of actions to complete an activity from the partial observations of the activity. We call this task ***action sequence forecasting***. This type of problems arise in practice, especially in robotics, e.g., robot assisted industrial maintenance, and assistive robotics in healthcare.

Our model observes only a handful of actions within a long activity. Then it forecasts the sequence of actions for the future without making any assumptions on the length of video or using precise future temporal annotations. In contrast to the majority of action anticipation and activity prediction models, ours is trained to predict *the future action sequence* as shown in figure 2. To solve this problem, there are several challenges that we need to tackle. First, our method needs to implicitly infer the goal of the person. Second, it should learn to what extent the person has completed the activity. Finally, it has to infer what other actions are needed to accomplish the activity. We formulate our solution such that all of this is learned in a data driven manner. Specifically, we make use of the complex relationship between observed video features and the future actions to learn a complex mapping between them. To facilitate that, we formulate it as a neural machine translation problem where the input is an observed RGB video and the target is a symbolic *sequence of future actions*. Specifically, we use a recurrent encoder-decoder architecture.

Each future action depends on the past observed feature sequence and interestingly, some of the observed features are important in determining the future actions more than others. For example, if our model predicts "adding sugar" as the future action, then it is more likely that our model gives a higher attention weight to frames having a cup or a mug. Therefore, we make use of an attention mechanism that allows us to align-and-attend past features when generating future actions. For each predicted future action, the attention mechanism processes the entire input feature sequence and selects the relevant set of observed frames that contain information about the future actions. Similar ideas have been explored before in other application domains, e.g., handwritten mathematical expression recognition [11]. Furthermore, our GRU encoder allows us to better model the temporal evolution of observed human actions and encode them into a temporally coherent hidden representation. The attention mechanism query these hidden temporal representations and provide useful information to the decoder GRU to generate accurate future actions.

Furthermore, the uncertainty of predictions increases with two factors; (1) the amount of data the model observes, and (2), how far into the future it predicts. If our model observes more data, perhaps the predictions are likely to be reliable. Moreover, if the model predicts far into the future, then predictions are likely to be unreliable. We develop a novel loss function that allows us to consider these two factors and extend the traditional cross-entropy loss to cater for these uncertainties. We also make use of optimal transport loss which allows us to tackle the exposure bias issue of this challenging sequence-to-sequence machine translation problem. Exposure bias arises when we use cross-entropy loss to train neural machine translation models where it provides an individual action-level training loss (ignoring the sequential nature) which may not be suitable for our task. The optimal transport loss is a more structured loss that aims to find a better matching of similar actions between two sequences, providing a way to promote semantic and contextual similarity between action sequences. In particular, this is important when forecasting future action sequences from observed temporal features.

Finally, we propose a model to predict action labels for future frames in a weakly supervised manner. Weakly supervised action forecasting is useful for practical applications, specially when it is harder to obtain frame level annotations (or start, end of actions). Specifically, we extend our action sequence forecasting model to perform action forecasting for future frames. Our weakly supervised model generates *pseudo representations* for future frames and then uses an attention mechanism to align them with forecasted future symbolic action sequence. Using this mechanism it predicts labels for future unseen frames. Our weakly supervised action forecasting method uses a novel GRU encoder-decoder architecture and we train this model using coarse action sequences (labels). Our model obtains results that are comparable to supervised methods and sometimes even outperforms them.

In a summary, our contributions are as follows:

- We propose an action sequence forecasting model that only utilizes the observed input frame sequence. Our architecture for weakly supervised action forecasting allows us to train with coarse annotations and predict action labels for frames at test time.
- We propose new loss functions that handle the uncertainty in future action sequence forecasting and we demonstrate the usefulness of optimal transport and the uncertainty losses.
- We extensively evaluate our method on four challenging action recognition benchmarks and obtain state of the art results for action forecasting. Our weakly supervised method obtains comparable results to prior supervised methods and in some datasets even outperforms them.

II. Related work.

We categorize the related work into three, 1. early action prediction and anticipation, 2. activity prediction and 3. weakly supervised action understanding and 4. machine translation.

Early action prediction and anticipation: Early action prediction aims at classifying the action as early as possible from partially observed action video. Typically, experiments are conducted on well segmented videos containing a single human action. In most prior work, methods observe about 50% of the video and then predict the action label [12], [13], [7]. In particular these methods can be categorized into four types. Firstly, some generate features for the future and then use classifiers to predict actions using generated features [14], [15]. Feature generation for future action sequences containing a large number of actions is a challenging task and therefore, not feasible in our case. Secondly, [7], [16], [17] develop novel loss functions to cater for uncertainty in the future predictions. Our work also borrows some concepts from these methods to develop loss functions but ours is applied over the future action sequence in contrast to applying over an action. Thirdly, some anticipation methods generate future RGB images and then classify them into human actions using convolution neural networks [18], [19]. However, generation of RGB images for the future is a very challenging task specially for longer action sequences. Similarly, some methods aim to generate future motion images and then try to predict action for the future [20]. However, we aim to forecast action sequences for unseen parts of the human activity and are more challenging than action anticipation. Therefore, action anticipation methods can not be used to solve our problem. Furthermore, our approach uses coarse level video annotations during training which allows someone to scale up our method for very large scale problems as-well-as for domains where obtaining frame level annotations is difficult.

Activity prediction: Some action forecasting methods assume that the number of future frames is given and predict the action label for each future frame [3], [9], [21]. These methods ([3], [9], [21]) require precise temporal annotations at frame-level during training and some methods use ground truth labels for observed actions [9] during inference. Several action forecasting methods are evaluated in [3] including CNN and RNN based approach. In particular this method [3] uses frame level annotations of observed data to train a frame classification model. This model predicts actions of the observed frames. Then using the generated actions of the observed frames, it directly predicts the actions for the future frames. Furthermore, the CNN, RNN methods of [3] predicts future actions for frames in a segment-wise manner until the desired prediction level is reached. In contrast, our model does not make any assumptions about the length of the sequence and rely on the model to emit the end-of-sequence token when predicting future action sequences. Furthermore, when predicting action labels for the future frames in a weakly supervised manner, our method generates pseudo representations for future frames and predicts action labels per-frame using the attention mechanism. Method in [9] uses a memory network which is also an encoder-decoder method. However, our model architecture and the loss functions used are different from [9]. Our model does not use frame level annotations and use only coarse video level annotations during training. In contrast, the method in [9] uses frame level annotations and frame features during training and even frame level annotations of observed frames during testing. First, both the observed frame sequence and the action sequence are encoded with a LSTM unit and then they are concatenated and fed to a memory network decoder to generate future actions. In contrast, our weakly supervised action forecasting model relies on a new encoder-decoder architecture with three dedicated decoders to generate future action labels for frames in a weakly supervised manner. We use only coarse action labels during training and our model does not need any annotated frames during inference as in [9]. Model presented in [21] consists of two parts, i.e., temporal feature attention module, and time-conditioned skip connection module for action forecasting. Our model is different from all these methods due to the differences in model architecture and type of supervision used. We aim to predict the future sequence of action (e.g. wash $\succ$ clean $\succ$ peel $\succ$ cut) and assign a label for each future frame in a weakly supervised manner without using frame level annotations during training.

Some activity prediction methods aim at predicting the next action in the sequence [10], [22] or focus on first person human actions [23], [24]. Specifically, [10] used to predict the next action using the previous three actions using motion, appearance, and object features with a two layered stacked LSTM. Authors in [22] use stochastic grammar to predict the next action in the video sequence. Even though these methods can be extended to predict the sequence of actions by recursively applying them, we face two challenges. Firstly, errors may propagate making future actions more wrong, and secondly it may not know when to stop producing action symbols, which is important when the actions are part of some larger activity. Sequence-to-sequence machine translations are naturally able to address both these two issues [25]. We make use of this strategy to solve our problem.

Weakly supervised action understanding: To the best of our knowledge we are the first to present a model for weakly supervised action forecasting. Weakly supervised methods are used for tasks such as action detection [26], [27] and segmentation [28]. Authors in [28] utilize weak annotations for action classification by aligning each frame with a label in a recurrent network framework. Specifically, they use connectionist temporal classification architecture with dynamic programming to align actions with frames. Authors in [29] learn to find relevant parts of an object/action after randomly suppressing random parts of images/videos. This method focuses only on very few discriminative segments. Authors in [30] directly predict the action boundaries using outer-inner-contrastive loss in a weakly supervised manner. Authors in [26] propose two terms that minimize the video-level action classification error and enforce the sparsity when selecting segments for action detection. Authors in [31] use attention-based mechanism to identify relevant frames and apply a pairwise video similarity constraint in a multiple instance framework. Our weakly

supervised model is different from what has been used in prior work with respect to two reasons. First, our model generates features for unseen frames and then assigns a label for each using only coarse sequence annotations as shown in Fig 2. Secondly, our network architecture contains a hierarchy of GRU encoders and decoders. To be precise, it consists of a single encoder and three decoders dedicated to predict and assign action labels for future frames.

Machine translation. Our method is also related to machine translation methods [25], [32], [33], [34]. However, none of these works use machine translation for action sequence forecasting from videos. Typically, machine translation is used for language tasks [25], [32]. To the best of our knowledge, we are the first to use neural machine translation for translating a sequence of RGB frames (a video) to a sequence of future action labels with *weak supervision*. Indeed, machine translation has been used for unsupervised learning of visual features [35] in prior work which is related to us. But they did not use it for predicting future action sequences.

III. Future action sequence generation.

A. Problem

We are given a video in which a human is performing an activity. Our model only observes the initial part of the video containing initial sequence of actions. The objective of this work is to train a model to predict the future unseen sequence of actions. A visual illustration of this model is shown in figure 2. Let us denote the observed RGB video by $X^o = \langle x_1^o, x_2^o, x_3^o, \cdots, x_p^o \rangle$ where x_p^o is the p^{th} frame. The observed action sequence is denoted by $Y^o = \langle y_1^o, y_2^o, \cdots y_{\mathcal{P}}^o \rangle$ (note that $p \neq \mathcal{P}$) and the future unseen ground truth action sequence by $Y^u = \langle y_1^u, y_2^u, \cdots y_{\mathcal{N}}^u \rangle$ where each action $y \in \mathcal{Y}$ and $\mathcal{Y}$ is the set of action classes and the start time of each action y_i is before or equal to the start time of y_{i+1}. In contrast to most other action forecasting problems, we do not use frame level annotations. Each observed (y_t^o) or future unseen action (y_t^u) may span over multiple frames and we do not explicitly use the start and end time of each action. However, during inference we are able to predict an action label for future unseen frames or explicitly infer the start and end of each future action.

First, we present our default future action sequence forecasting model in section III-B. Second, we extend this model to infer start and end of future actions only using coarse labels sequences Y^o and Y^u for training. This weakly supervised action forecasting method is presented in section IV. Our fully supervised model which uses frame level annotations for observed and future actions is presented in IV-A.

B. Future action sequence forecasting model

In contrast to other action forecasting methods that operates at frame level (or clip level), we do not know the label of each observed RGB frame x_p^o. Our model has access to frame sequence X^o only. We train a model $\phi(, \Theta)$ that predicts unseen action sequence Y^u from seen RGB feature sequence X^o where Θ are the parameters of model, i.e. $Y^u = \phi(X^o, \Theta)$. We do not make use of ground-truth action sequence Y^o during training or inference. Therefore, our method does not need any frame level action annotations as in prior action forecasting methods [3], [9].

We formulate this problem as a sequence-to-sequence machine translation problem [25], [32], [33], [34] where we use observed rgb sequence X^o as the input sequence. Then the symbolic unseen action sequence Y^u is the target sequence. Specifically, we use an GRU-based encoder-decoder architecture. Our hypothesis is that the encoder-decoder machine translation would be able to learn the complex relationship between seen feature sequence and future actions. To further improve the model predictive capacity, we also use attention over encoder hidden state when generating action symbols for the future and use novel loss functions to tackle uncertainty. Next we describe our model in detail.

1) GRU-encoder-decoder: We use GRU-based encoder-decoder architecture for translating video sequence into future action sequence. Our encoder consists of a bi-directional GRU cell. Let us define the encoder GRU cell which takes the observed feature sequence consisting of p elements as input. We define the encoder GRU by $f_e()$ for time step t as follows:

$$\overrightarrow{\mathbf{h}}_t, \overleftarrow{\mathbf{h}}_t = f_e(\mathbf{x_t^o}, \overrightarrow{\mathbf{h}}_{t-1}, \overleftarrow{\mathbf{h}}_{t-1}) \tag{1}$$

where $\overrightarrow{\mathbf{h}}_t, \overleftarrow{\mathbf{h}}_t \in R^{\mathcal{D}}$ are the forward and backward hidden states at time t. The initial hidden state of the encoder GRU is set to zero. Then we make use of a linear mapping $W_e \in R^{2\mathcal{D}\times\mathcal{D}}$ to generate a unified representation of both forward and backward hidden states for each time step $\overrightarrow{\mathbf{h}}_t$, $\overleftarrow{\mathbf{h}}_t$ as follows:

$$\mathbf{h_t} = [\overrightarrow{\mathbf{h}}_{t-1}, \overleftarrow{\mathbf{h}}_{t-1}] \times W_e \tag{2}$$

where $[\cdot, \cdot]$ indicates the concatenation of forward and backward hidden states. Therefore, the outcome of the encoder GRU is a sequence of hidden state vectors denoted by $\mathbf{H} = \langle \mathbf{h_1^o}, \mathbf{h_2^o}, \cdots \mathbf{h_p^o} \rangle$. The bi-directional GRU encode more contextual information which might inherently enable the model to infer the intention of person doing the activity. The decoder is a forward directional GRU $f_d()$, that generates the decoder hidden state $\mathbf{g_q} \in R^{\mathcal{D}}$ at decoding time step q define as follows:

$$\mathbf{g_q} = f_d([\mathbf{c_{q-1}}, \hat{\mathbf{y}}_{\mathbf{q-1}}], \mathbf{g_{q-1}}) \tag{3}$$

where $\hat{\mathbf{y}}_{\mathbf{q-1}}$ is the predicted target action class score vector at step q-1. The input to decoder GRU $f_d()$ at time step is a concatenation of the context vector $\mathbf{c_{q-1}}$ and the previously predicted action score vector $\hat{\mathbf{y}}_{\mathbf{q-1}}$ denoted by $[\mathbf{c_{q-1}}, \hat{\mathbf{y}}_{\mathbf{q-1}}]$. We obtain the action score vector at step q of the decoder using following linear mapping:

$$\hat{\mathbf{y}}_{\mathbf{q}} = \mathbf{g_q} \times U \tag{4}$$

where $U \in R^{\mathcal{D}\times|\mathcal{Y}|}$ is a learnable parameter. Note that the output symbol at step q of the decoder is obtain by `argmax` operator, i.e., $\hat{y}_q = \texttt{argmax}\ \hat{\mathbf{y}}_{\mathbf{q}}$. The decoder is initialized by the final hidden state of the encoder (i.e. $\mathbf{g_0} = \mathbf{h_p^o}$ where $\mathbf{h_p^o}$ is the final hidden state of the encoder). The initial symbol of the decoder is set to SOS (start of sequence symbol) during training and testing. The decision to include the previous predicted action $\hat{\mathbf{y}}_{\mathbf{q-1}}$ as an input in the decoder is significant

as now the decoder model has more semantic information during the decoding process. One choice would be to simply ignore the previously predicted action symbol. However, that would hinder the predictive capacity of the decoder as decoder is not explicitly aware of what it produced in the previous time step. Conceptually, now the decoder is trying to find the most likely next symbol $P(y_q|y_{q-1}, \mathbf{g_{q-1}}, \mathbf{c_{q-1}})$ using both previous symbol and the contextual information.

Next we describe how to generate the context vector $\mathbf{c_{q-1}}$ which summarizes the encoder-decoder hidden states using attention mechanism.

2) Attention over encoder hidden state: It is intuitive to think that not all input features contribute equally to generate the output action symbol $\hat{y}_q$ at decoder step q. Therefore, we propose to make use of attention over encoder hidden states $\mathbf{H}$ to generate the context vector $\mathbf{c_{q-1}}$ which serves as an part of input to the decoder GRU. Specifically, to generate $\mathbf{c_{q-1}}$, we linearly weight the encoder hidden vectors $\mathbf{H} = \langle \mathbf{h_1^o}, \mathbf{h_2^o}, \cdots \mathbf{h_p^o} \rangle$, i.e.,

$$\mathbf{c_q} = \sum_i \frac{exp(\alpha_i^q)}{\sum_j exp(\alpha_j^q)} \mathbf{h_i^o} \tag{5}$$

where α_i^q is the weight associated with the encoder hidden state $\mathbf{h_i^o}$ to obtain q-th context vector defined by the following equation.

$$\alpha_i^q = tanh([\mathbf{h_i^o}; \mathbf{g_q}] \times W_{att}) \times V \tag{6}$$

Here $W_{att} \in R^{2\mathcal{D} \times \mathcal{D}}$ and $V \in \mathcal{D}$ are learnable parameters and α_i^q depends on how well the encoder-decoder hidden states $\mathbf{h_i^o}, \mathbf{g_q}$ are related. This strategy allows us to attend all encoder hidden states $\mathbf{H} = \langle \mathbf{h_1^o}, \mathbf{h_2^o}, \cdots \mathbf{h_p^o} \rangle$ when generating the next action symbol using decoder GRU. During training we make use of the teacher forcing strategy to learn the model parameters of the encoder-decoder GRUs where we randomly choose to use $\mathbf{y_q}$ instead of $\hat{\mathbf{y}}_\mathbf{q}$ in equation 3 with a probability of 0.5. This is to make sure that the inference strategy is not too far away from the training strategy and that convergence takes place faster. During inference, given the input features sequence, we pass it thorough the encoder-decoder to generate future action sequence until we hit the end-of-sequence symbol (EOS). The model is also trained with start-of-sequence (SOS) symbol and EOS.

3) Tackling the uncertainty: Correctly predicting the future action sequence from a partial video is challenging as there are more than one plausible future action sequences. This uncertainty increases with respect to two factors; 1. to what extent we have observed the activity, (the more we observe, the more information we have to make future predictions) and 2. how far into the future we are going to predict using observed data (if we predict too far into the future, there are more possibilities and more uncertainty). To tackle these two factors, we propose to modify the cross-entropy loss which is typically used in sequence-to-sequence machine translation[1]. Let us assume that we have observed $\mathcal{P}$ number of actions, and we are predicting a total of $\mathcal{N}$ number of action symbols. Let us denote the cross-entropy loss between the prediction $(\hat{\mathbf{y}}_\mathbf{q})$ and the ground truth $(\mathbf{y}_q)$ by $\mathcal{L}(\hat{\mathbf{y}}_\mathbf{q}, \mathbf{y}_q)$. Then our novel loss function that handles the uncertainty $(L_{un}(\hat{Y}^u, Y^u))$ for a given video X^o, Y^u is define by

$$L_{un} = (1 - exp(-\frac{\mathcal{P}}{\mathcal{N}})) \sum_{q=1}^{\mathcal{N}} exp(-q) \mathcal{L}(\hat{\mathbf{y}}_\mathbf{q}, \mathbf{y}_q) \tag{7}$$

where the term $(1 - exp(-\mathcal{P}/\mathcal{N}))$ takes care of shorter observations and makes sure that longer action observations contributes more to the loss function. If the observed video contains less actions (information), then predictions made by those are not reliable and therefore does not contribute much to the overall loss. Similarly, the second inner term $exp(-q)\mathcal{L}(\hat{\mathbf{y}}_\mathbf{q}, \mathbf{y}_q)$ makes sure that those predictions too far into the future make only a small contribution to the loss. If our model makes a near future prediction, then possibly model should do a better job and if it makes an error, we should penalize more. During training, we make use of sequential data augmentation to better exploit the above loss function. In-fact, for given a training video consist of $\mathcal{M}$ actions (i.e. $Y = \langle y_1, y_2, \cdots y_\mathcal{M} \rangle$), we augment the video to generate $\mathcal{M} - 1$ observed sequences where $Y^o = \langle y_1^o, y_2^o, \cdots y_t^o \rangle$ and $Y^u = \langle y_{t+1}^o, \cdots y_M^o \rangle$ for $t = \{1, \cdots, M-1\}$. Then we train our networks with these augmented video sequences with the uncertainty loss.

4) Optimal Transport Loss (OT): So far, the cross-entropy loss is applied over actions in a point-wise manner without taking into account the topological or the geometric structure of action space. The element-wise cross-entropy loss obtained for action at step-q of the decoder only relies on the ground-truth action at step-q and it does not take the sequence-to-sequence structural nature of the task. Moreover, when the predicted sequence is longer than the ground truth, the cross-entropy loss requires adhoc end-of-sequence token (class) to handle this. However, ideally, the encoder-decoder model should be able to predict the target action sequence of the future by considering structural nature of this task.

The optimal transport defines a distance measure between probability distributions over a metric space by considering the topology, and in our case the topology of actions sequences. It is desirable to exploit this metric structure in the action sequence space using optimal transport loss [36] over predicted action sequences. We propose to make use of optimal transport loss of [36] defined by

$$D_c(\mu, \nu) = \inf_{\gamma \in \Pi(\mu, \nu)} \mathbb{E}_{(x,y) \sim \gamma}[c(\mathbf{x}, \mathbf{y})] \tag{8}$$

where $\Pi(\mu, \nu)$ is the set of all joint distributions $\gamma(\mathbf{x}, \mathbf{y})$ with marginals $\mu(\mathbf{x})$ and $\nu(\mathbf{y})$ and $c(\mathbf{x}, \mathbf{y})$ is the cost function for moving $\mathbf{x}$ to $\mathbf{y}$ in the sequence space. We take the cost function to be the L2 norm i.e. $c(\mathbf{x}, \mathbf{y}) = \|\mathbf{x} - \mathbf{y}\|_2$.

Specifically, we consider the optimal transport distance between two discrete action distributions $\mu, \nu \in \mathbf{P}(\mathbb{A})$ of the action sequences where $\mathbb{A}$ is the action space. The discrete distributions μ, ν can be written as weighted sums of Dirac delta functions i.e. $\mu = \sum_{i=1}^n \mathbf{u}_i \delta_{\mathbf{x}_i}$ and $\nu = \sum_{j=1}^m \mathbf{v}_j \delta_{\mathbf{y}_j}$ with $\sum_{i=1}^n \mathbf{u}_i = \sum_{j=1}^m \mathbf{v}_j = 1$. Given a cost matrix $\mathbf{C} \in \mathbb{R}_+^{n \times m}$

[1]This strategy may be applicable to other loss functions as well.

where $\mathbf{C}_{ij}$ is the cost from $\mathbf{x}_i$ to $\mathbf{y}_j$, the optimal transport loss is equivalent to

$$L_{ot}(\mu,\nu) = \min_{\mathbf{P}\in\Pi(\mathbf{u},\mathbf{v})} \sum_{i,j} \mathbf{P}_{ij}\mathbf{C}_{ij} \tag{9}$$

where $\Pi(\mathbf{u},\mathbf{v}) = \{\mathbf{P}\in\mathbb{R}_{+}^{n\times m}\}|\mathbf{P}\mathbf{1}_m = \mathbf{u}, \mathbf{P}^\top\mathbf{1}_n = \mathbf{v}\}$ and $\mathbf{1}_n$ is a n-dimensional vector of all ones.

The minimum P^* in equation 9 is the ideal optimal transport solution that caters for the topological structure of predicted and ground truth action sequences. The cost function is defined in the Euclidean space over action predictions as follows:

$$C_{ij} = ||\hat{\mathbf{s_i}} - \mathbf{s_j}||_2 \tag{10}$$

where $\hat{\mathbf{s_i}}$ is the predicted action score vector at step i and $\mathbf{s_j}$ is the one-hot-vector obtained from the ground truth action sequence Y^u at step j. Note that both $\hat{Y}^u$ and Y^u are discrete action symbol sequences and the optimal transport loss is complementary to the cross entropy loss and vice-versa.

Because the optimal transport assignment problem is formulated as a permutation problem, we make use of the Sinkhorn divergence method proposed in [37] to estimate the optimal transport loss in equation 9. Using this Sinkhorn algorithm implementation proposed in [37], we compute the optimal transport loss between the predicted and ground-truth action probability distributions from equation 9. Let S be the $\mathcal{N}\times|\mathcal{Y}|$ ground truth tensor where S_{jk} contains the probability value of action k in step j (i.e. each row j is equal to one-hot vector $\mathbf{s_j}$), and $\hat{S}$ be the corresponding tensor containing the predicted probability values (i.e. each row i is equal to probability vector $\hat{\mathbf{s_i}}$). The optimal transport loss (L_{ot}) computed using the Sinkhorn algorithm is then denoted by $L_{sh}(\hat{S},S)$. The combination of both losses is given by the following:

$$L_{total} = L_{un}(\hat{Y}^u, Y^u) + \beta \times L_{sh}(\hat{S}, S), \tag{11}$$

where β is the trade-off parameter and $L_{un}(\hat{Y}^u, Y^u)$ is obtained by equation 7.

IV. Weakly Supervised Future Action Forecasting

In this section we present a method to infer the temporal extent of each future action using only the coarse labels of observed and unseen future action sequences. Therefore, essentially our method is a weakly supervised action forecasting method. Specifically, during training we make use of observed feature sequence $X^o = \langle x_1^o, x_2^o, x_3^o, \cdots, x_p^o\rangle$, observed action sequence $Y^o = \langle y_1^o, y_2^o, \cdots y_{\mathcal{P}}^o\rangle$ and the future unseen action sequence $Y^u = \langle y_1^u, y_2^u, \cdots y_{\mathcal{N}}^u\rangle$. Note that both Y^o and Y^u are coarse sequences, i.e. we do not know the temporal extent of each action. Both Y^o and Y^u are used to compute the loss during training. At test time, we predict future action sequence $\hat{Y^u}$ and a label of for each unseen frame only using observed feature sequence X^o. To do that, we use the attention mechanism presented in section III-B2. High level illustration of our novel architecture is shown in Fig. 3. Now we give more details of our weakly supervised action forecasting method

First, our model processes the feature sequence X^o using a GRU encoder $f_e^o()$ to obtain a hidden state sequence H^o of the same length as X^o.

$$H^o = f_e^o(X^o) \tag{12}$$

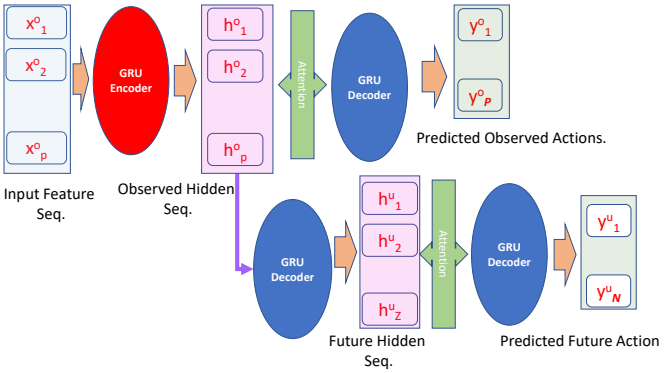

Fig. 3. A visual illustration of our weakly supervised action forecasting architecture. It consists of a encoder and three decoder GRUs. The first decoder uses attention mechanism to align observed hidden sequence H^o with the observed action sequence Y^o. The second GRU decoder processes last observed hidden state h_p^o and then decodes to generate pseudo states for the future features denoted by $H^u = \langle h_1^u, \cdots, h_Z^u\rangle$. The attention mechanism is used to align predicted future action sequence $\hat{Y^u}$ with pseudo hidden states H^u. This allows us to estimate a label for each future hidden state using the attention weights and action scores $< \mathbf{y_1^u}, \cdots . \mathbf{y_N^u} >$.

Then, a GRU decoder $f_d^o()$ with attention is used to decode H^o to obtain the observed action sequence $\hat{Y^o}$. The encoder-decoder with attention used here (i.e. $f_e^o()$ and $f_d^o()$) is explained in section III-B1 and III-B2.

The second GRU decoder $f_d^u()$ **without attention** takes the last hidden state h_p^o of H^o as the initial hidden state and decodes to generate a sequence of future hidden states for unseen frames. Let us assume there are Z number of future unseen frames. Therefore, $f_d^u()$ generates the future hidden state sequence $H^u = \langle h_1^u, \cdots h_Z^u\rangle$ as follows:

$$H^u = f_d^u(h_p^o). \tag{13}$$

We call these future unseen hidden vectors as **pseudo states** and in practice Z is known or given to us. The final GRU decoder (f_d^{uy}) with attention is applied over these pseudo state sequence H^u to obtain future action sequence $\hat{Y^u}$ as follows:

$$\hat{Y^u} = f_d^{uy}(H^u). \tag{14}$$

The attention allows us to align each future hidden state h_t^u with the corresponding future label $\hat{y_q^u}$ of $\hat{Y^u}$. Similar to equation 3, inputs to this decoder are the predicted future action score and the context vectors. Let α_t^q be the attention score on t^{th} unseen hidden state h_t^u for generating q^{th} action symbol $\hat{y_q^u}$ of $\hat{Y^u}$. The attention scores are obtained as explained in section III-B2. The attention score indicates the contribution of future frame x_t^u for generating action y_q^u of the future. Therefore, we can assign action score s_t for each future frame x_t^u using the following equation

$$s_t = \sum_q \alpha_t^q \times \mathbf{y_q^u} \tag{15}$$

where $\mathbf{y_q^u}$ is the score vector of q^{th} future action symbol of $\hat{Y^u}$. Even though, we have never observed any of future features, this formulation allows us to generate pseudo hidden states for future and then decode that to generate future action sequence. This attention mechanism allows us to assign a label for each

future frame without using any explicit frame level annotation during training.

To train $f_e^o(), f_d^o(), f_d^u(), h_d^{uy}()$ we use combination of two losses as follows:

$$Loss = \mathcal{L}(Y^o, \hat{Y^o}) + \gamma \mathcal{L}(Y^u, \hat{Y^u}) \tag{16}$$

where the loss functions $L()$ are explained in section III-B3. As before, we use end-of-sequence token (EOS) during training and testing for both $\hat{Y^o}$ and $\hat{Y^u}$. In our implementation we make sure that the feature dimensions of X, H^o, H^u are the same.

A. Fully supervised action forecasting

We also extend the method presented in section IV for fully supervised action forecasting. In this case, the observed and future action sequences Y^o, Y^u are frame specific and the loss function in Eq. 16 is applied over frame level action annotations during training. However, at inference, model takes observed feature sequence X^o as input and predicts action labels for each future frame. We do not use EOS token for fully supervised model.

V. Experiments.

In this section we extensively evaluate our model using three challenging action recognition datasets, namely the Charades[38], MPII Cooking[39] and Breakfast[40] datasets. Next we give a brief introduction to these datasets. MPII-Cooking Dataset has 65 fine grained actions, 44 long videos with a the total length of more than 8 hours. Twelve participants interact with different tools, ingredients and containers to make a cooking recipe. We use the standard evaluation splits where total of five subjects are permanently used in the training set. Rest six of seven subjects are added to the training set and all models are tested on a single subject and repeat seven times in a seven fold cross-validation manner. In this dataset, there are 46 actions per video on average.

Charades dataset has 7,985 video for training and 1,863 videos for testing. The dataset is collected in 15 types of indoor scenes, involves interactions with 46 object classes and has a vocabulary of 30 verbs leading to 157 action classes[38]. On average there are 6.8 actions per video which is much higher than other datasets having more than 1,000 videos.

Breakfast dataset[40] consist of 1,712 video where 52 actors making breakfast dishes. There are 48 fine-grained actions classes and four splits. On average each video consists of 6.8 actions per video.

There are no overlapping actions in Breakfast and Cooking datasets. Charades has handful of videos with overlapping actions. To generate ground truth action sequences, we sort the list of actions by start times, ignoring the end time of the actions.

A. Performance evaluation measures:

We measure the quality of generated future action sequences using BLEU-1 and BLEU-2 scores[41]. These are commonly used in other sequence evaluation tasks such as image captioning. We use the standard BLEU score definition proposed in the Machine Translation and Natural Language Community[41] which is also publicly implemented in Python nltk toolbox. We also report sequence-item classification accuracy which counts how many times the predicted sequence elements match the ground truth in the exact position. Furthermore, we also report the mean average precision (mAP) which does not account for the order of actions. To calculate mAP, we accumulate the action prediction scores of the unseen video and compare it with the ground truth. BLEU-1, BLEU-2 and sequence-item classification accuracy reflects the sequence forecasting performance while the mAP only accounts for holistic future action classification performance discarding the temporal order of actions.

The use of BLEU scores is somewhat novel in action forecasting. In machine translation, BLEU score is used to compare a candidate sequence of words against a reference translation. Similarly, we use BLEU scores in the context of action sequence forecasting to provide a precision measure over action sequences. For example, BLEU-2 score indicates the precision of each models' ability to correctly predicts two-action compositions (e.g open $\succ$ close, wash $\succ$ peel). Therefore, BLEU scores provides complementary information to sequence-item classification accuracy.

B. Feature extraction and implementation details:

Unless specifically mentioned, we use effective I3D features [42] as the video representation for all datasets. First, we fine-tune I3D networks for video action classification using provided video level annotations. Afterwards, we extract 1024-dimensional features to obtain a feature sequence for each video.

C. Evaluating our model

In the following sections we evaluate various aspects of our model aiming to provide some insights to the reader. First, we evaluate our action sequence forecasting model from section V-D to V-G. In section V-D we evaluate our action sequence forecasting model with cross-entropy loss. Then in section V-E we evaluate the impact of new loss functions for action sequence forecasting. After that in section V-F we compare our model with baselines models for predicting the next action. We also evaluate the performance of our model when predicting the next action conditioned on the last three actions in section V-G. Finally, we evaluate weakly supervised action forecasting in section V-H and compare with other published methods.

D. How well does it perform in action sequence forecasting?

In this section we evaluate our action sequence forecasting model using all three datasets. During training, for each given video X and the action sequence $Y = \langle y_1, y_2, \cdots, y_N \rangle$, our model take feature sequence X^o corresponding to observed action sequence $Y^o = \langle y_1^o, \cdots, y_i^o \rangle$ and then predict future action sequence $Y^u = \left\langle y_{i+1}^u, \cdots, y_N^u \right\rangle$ for all i values (i.e. for $i = 1, \cdots, N-1$). The observed i^{th} action symbol is

Fig. 4. Qualitative results obtained with our method on MPII Cooking dataset. Correctly predicted actions are shown in green and the wrong ones in red.

TABLE I
GRU ENCODER-DECODER PERFORMANCE ON ACTION SEQUENCE FORECASTING

Dataset	Setup	BLEU-1 (%)	BLEU-2 (%)	Seq. Item. Acc (%)	mAP
Charades	Random	1.04	0.35	0.28	4.40
Charades	Classification	15.26	2.78	5.35	28.40
Charades	Forecasting (Mean+GRU)	5.15	1.87	2.30	5.90
Charades	Forecasting (GRU-ED)	5.75	2.18	1.53	5.10
Charades	Forecasting (GRU-ED Att.)	7.95	2.87	2.60	6.10
Breakfast	Random	1.33	0.49	0.70	7.53
Breakfast	Classification	51.83	37.38	26.35	46.89
Breakfast	Forecasting (Mean+GRU)	25.65	10.23	18.11	24.94
Breakfast	Forecasting (GRU-ED Att.)	34.56	21.15	21.29	30.24
MPII-Cooking	Random	1.28	0.48	0.47	6.53
MPII-Cooking	Classification	25.74	14.34	14.86	20.60
MPII-Cooking	Forecasting (GRU-ED Att.)	8.70	4.10	4.50	10.80

denoted by y_i^o, and corresponds to a real action e.g. "opening fridge". We use this action sequence sampling strategy to evaluate test videos for all possible i values. Unless otherwise specified, we use this strategy for training and testing which we call as the **Action Sequence Forecasting Setup**. With this augmentation strategy, we obtain much larger dataset for training and evaluation. This setup is different from what has been done in prior work[3].

We report results using our GRU-based encoder-decoder model trained with attention (GRU-ED Att.) and traditional cross-entropy loss for action sequence forecasting. As a baseline, we report results for random performance. In this case, for a given video, we randomly generate the next score vector to obtain the next action symbol for the unseen sequence. As the second baseline, we process the entire video feature sequence to obtain the full action sequence denoted by **Classification Setup**. In this case, we observe the feature sequence X^o corresponding to all actions and then output the action sequence $Y = \langle y_1, y_2, \cdots, y_N \rangle$. We do this using the same model presented in section III-B and cross-entropy loss. Results obtained by sequence classification model serves as a soft upper bound for the action sequence forecasting model.

Additional baselines: To validate the effectiveness of attention mechanism, we also compare results with GRU encode-decoder without attention denoted by GRU-ED. To validate the effectiveness of encode-decoder architecture, we also report results using another GRU baseline where the input to GRU is the mean I3D feature. This model is denoted by Mean+GRU.

Results are shown in Table I. We make several observations. First, our model performs significantly better than the random performance. Sequence item classification accuracy (which is a strict measure) reflects the difficulty of action sequence forecasting task. In the forecasting setup, we obtain item classification accuracy of 2.60, 4.50, and 21.29 where the random performance is 0.28, 0.47, and 0.70 on Charades, MPII cooking and Breakfast respectively. The random performance indicates the difficulty of forecasting task. Our model is 10-30 times better than random performance.

The difference in results between classification and forecasting setups is not too drastic, especially for Breakfast and Charades. Our classification model obtains seq. item accuracy of 5.35 while our forecasting model reach 2.60 on Charades. Similarly for MPII cooking dataset, the classification model obtains 14.86 and our action forecasting model's performance is 4.50. Interestingly, seq. item classification accuracy of 26.35 and 21.29 is obtained for classification and forecasting models

TABLE II
EVALUATING THE IMPACT OF UNCERTAINTY LOSSES AND THE OPTIMAL TRANSPORT LOSS. ACC. IS THE SEQUENCE ITEM CLASSIFICATION ACCURACY.

Loss	BLEU-1	BLEU-2	Acc. (%)	mAP (%)
Charade dataset.				
Cross-entropy	7.95	2.87	2.6	6.1
L_{un}-past-only	8.11	2.98	2.6	6.1
L_{un}-future-only	8.61	3.11	2.8	6.4
L_{un}-both	8.80	3.30	2.9	7.2
L_{ot}	7.73	3.06	3.4	7.2
L_{ot} + L_{un}-future-only	**9.59**	**3.92**	**4.0**	**8.2**
MPII Cooking dataset.				
Cross-entropy	8.70	4.10	4.50	10.80
L_{un}-future-only	9.22	5.00	5.64	10.36
L_{ot}	8.20	4.75	6.15	11.30
L_{ot} + L_{un}-future-only	**11.43**	**6.74**	**8.88**	**12.04**

respectively on Breakfast. For action forecasting task, the Charades dataset is the most challenging and the least is Breakfast dataset. Interestingly, for BLEU-2, the classification model obtains 2.78 while future action forecasting model performs better on Charades dataset.

The encoder-decoder model without attention (GRU-ED) improves results over Mean+GRU model on BLEU-1 and BLEU-2 scores on the challenging Charades dataset. Interestingly, when attention mechanism is employed to the GRU encoder-decoder, the results improve over the Mean+GRU model on all metrics. BLEU-1 score is improved from 5.15 to 7.95 and BLEU-2 score from 1.87 to 2.87. Furthermore, sequence item classification accuracy is improved from 2.30 to 2.60. These results suggest the effectiveness of encoder-decoder architecture with attention for action forecasting on the most challenging dataset. On the Breakfast dataset, we see even massive improvements, where the GRU-ED Att. model obtains an improvement of 8.91 for BLEU-1 and 10.92 for BLEU-2. Similarly, we see an improvement of 3.18% for sequence item classification accuracy. We conclude that GRU encoder-decoder with attention is effective for future action sequence forecasting problem. These results indicate the effectiveness of our method for future action sequence forecasting task. However, these results also suggest that there is more to do. Later in the experiments, we show how to improve these results.

E. What is the impact of loss functions?

In this section we evaluate our method using the uncertainty and optimal transport loss functions for action sequence forecasting setup. The uncertainty loss consist of two parts in Equation 7.

1) the effect of the fraction of past observations ($1 - exp(\mathcal{P}/\mathcal{N})$) denoted by L_{un}*-past-only*.
2) the extent of future predictions ($exp(-q)$) denoted by L_{un}*-future-only*.

First, we analyze the impact of these two terms separately and then evaluate them jointly. We also demonstrate the impact of optimal transport loss alone (L_{ot}). Finally, we evaluate combination of all losses where we set the β of Equation 9 to be 0.001. Results are reported in Table II.

TABLE III
PERFORMANCE COMPARISON FOR PREDICTING NEXT ACTION. MLP IS THE MULTIPLE LAYERED PERCEPTRON.

	Charades		MPII Cooking		Breakfast	
Method	Acc. (%)	mAP	Acc. (%)	mAP	Acc. (%)	mAP
MLP	3.9	1.7	7.1	4.1	16.2	8.8
LSTM	2.5	1.3	2.4	3.0	4.3	3.0
Our	**6.8**	**3.0**	**11.0**	**9.2**	**16.4**	**11.8**

From the results in Table II, we see that both uncertainty and optimal transport losses are more effective than the cross-entropy loss which justifies our hypothesis about these new loss functions. Interestingly, the loss term(L_{un}-future-only), obtains the best results for BLEU scores while OT loss obtain best action sequence classification accuracy for an individual loss. The combination of two uncertainty losses perform better than individual ones. Combination of both L_{ot} and L_{un}-future-only perform much better than all others obtaining a significant improvement in BLEU-1 and BLEU-2 scores from best of 7.95 to 9.59 and 2.87 to 3.92 on Charades dataset. Similar trend can be seen for MPII-Cooking with consistent improvements. This shows that optimal transport loss and L_{un}-future-only are complimentary to each other. Though two uncertainty losses perform better than cross-entropy loss, unfortunately, the combination of all three losses do not seem to be useful. Perhaps we need a better way to combine both uncertainty losses with the OT loss which we leave for further investigation in the future.

We visualize some of the obtained results in figure 4. Interestingly, our method is able to generate quite interesting future action sequences. In the first example, our method accurately obtain four out of five actions. In the second example, it predicts two actions correctly, however the predicted action sequence seems plausible though it is not correct.

F. How does it work for predicting the next action?

.

In this section, we evaluate the impact of our sequence-to-sequence encoder-decoder architecture for predicting the next action. For a given observed sequence $Y^o = \langle y_1^o, \cdots, y_i^o \rangle$, the objective is to predict the next action y_{i+1}^u for all i values of the video. Once again y_i^o is the i-th action of the video. As before, we generate all train and test action sequences. For comparison, we also use two layered fully connected neural network (MLP) which applies mean pooling over the observed features and then use MLP as the classifier. Similarly, we also compare with a standard LSTM which takes the input feature sequence and then predict the next action only. For our method and two baselines (LSTM, MLP), we use the same hidden size of 512 dimensions. For all models, we use the same activation function, i.e. `tanh()`. We report results in Table III.

First, we see that MLP obtains better results than LSTM. Second, our sequence-to-sequence method with attention performs better than both LSTM and MLP methods. MLP obtains 1.7 mAP for predicting the next action indicating features do not contain enough information about future and more complicated mechanism is need to correlate past features

TABLE IV
ACTION FORECASTING PERFORMANCE FOR USING ONLY THE FEATURES FROM PREVIOUS THREE ACTIONS ON CHARADES DATASET.

Loss	Accuracy (%)	mAP (%)
Cross-entropy	3.54	1.7
L_{ot} + Cross-entropy	6.25	2.3

with the future action. Our method obtains far better results than these two baselines indicating the effectiveness of our sequence-to-sequence architecture for next action prediction task. We conclude our model is better suited for future action prediction than MLP and LSTM.

G. What if we only rely on three previous actions?

In this experiment we evaluate the performance of our model when we predict the next action using only the three previous actions. Here we train and test our method using all augmented action sequences. As before we use I3D features from the seen three actions and aims to predict the next action class. We also compare traditional cross-entropy with (L_{ot} + Cross-entropy) loss. Results are reported on Table IV.

First, even for our method, we see a drop in performance from the results reported in previous experiment in Table III. When we predict the next action using all previous action features, with the cross-entropy loss, we obtain a classification accuracy of 6.8% in Table III whereas, in Table IV, our cross-entropy method obtains 3.54% only. This suggests that it is better to make use of all available information from observed video features and just let the attention mechanism to find the best features. Secondly, the optimal transport loss combined with cross-entropy loss improve results indicating it is complimentary even in this constrained case. For this experiment there is no need to make use of uncertainty loss as there is only one action to predict.

H. Evaluating weakly supervised action forecasting and comparison to other SOA methods.

In this section we evaluate our weakly supervised action forecasting model presented in section IV. For this experiment we use Breakfast dataset and commonly used 50Salads dataset [43]. In all prior experiments, we focus on forecasting future action sequence whereas most recent methods in the literature take a somewhat different approach[3], [9], [21]. These methods observe $p\%$ of the video and aims to predict future actions for $q\%$ of the video assuming length of video is known and frame level action annotations (at least the start and end of each action is known) are provided. In this section we follow the protocol used in[3], [9], [21]. However, our weakly supervised method does not make use of any frame level annotations during training.

First, we compare our fully supervised method (section IV-A) against the weakly supervised method using I3D features on Breakfast dataset. We compare our results with[3], [21] and report mean per class accuracy as done in[3], [21]. Unfortunately, the method in [9] uses ground truth action sequence labels (the observed actions) during inference which is not a realistic setup. As a fair comparison with methods proposed in[3], [21], we also experiment with the Fisher Vector (FV) features used in[3], [21]. Results for Breakfast dataset are reported in Table V.

When we use I3D features, our supervised method outperforms all baselines presented in [3], [21] by a large margin, including larger prediction percentages such as 0.5. On average, our supervised method obtains an improvement of 4.6% over the best prior model in [21]. Specifically, the biggest average improvement is obtained when we observe only the 20% of video. In this case, the average improvement is 5.1% across all prediction percentages. We also see a consistent improvement over all (p%) percentages.

When we use I3D features, our *weakly supervised* method also outperforms fully supervised results of [3], [21] in majority cases. It fails only in two extreme cases, e.g., when predicting 50% into the future. This indicates the power of I3D features and the effectiveness of our weakly supervised method. With I3D, our weakly supervised method is only 3.8% behind our fully supervised method on average and in one instance it is only 0.3% behind fully supervised results (i.e. observe 30% and predict 10%). As our weakly supervised method does not use any frame level annotations, this is a positively surprising result. Even more conclusive trend can be seen when we use Fisher Vector features.

Most interestingly, our weakly supervised results are comparable to [3], [21] when we use Fisher Vector features (FV). Somewhat surprisingly, when predicting 10% to the future (p=10%), our weakly supervised method obtains better results than supervised methods of [3], [21] indicating the effectiveness of our weakly supervised model presented in section IV. Our weakly supervised method is only 1.4%, 0.6% behind our supervised and recent [21] methods respectively. Most interestingly, it is 0.1% better than supervised CNN method of [3]. Our weakly supervised method performs relatively better when we observe more data (i.e. results for 30% observation is comparable to supervised performance of [3], [21]). In three out of eight cases, our weakly supervised method (with FV) outperforms supervised state-of-the-art methods such as [3], [21]. We attribute this improvement to the model architecture and the attention mechanism.

Specifically, the use of bidirectional GRU helped to improve results. Bi-directional encoding allows us to better exploit temporal dependencies in feature sequence. Furthermore, in our implementation we make sure that the feature dimensions of X, H^o, H^u are the same –see section IV. We notice that lower or higher dimensions for H^o and H^u hinder the performance. One interesting question is weather one should enforce the distribution of future hidden features $P(H^u)$ correlate with unseen future features distribution $P(X^u)$? We leave this question as a future exploration.

We also notice that the loss function in Eq. 16 plays a special role. Specifically, the best results are obtained when we give slightly higher importance to the second term of this loss by setting γ to be 2.0. However, very large γ values (such as $\gamma = 5.0$ or $\gamma = 10.0$) are worse than smaller values (e.g. $\gamma = 1.0$). Interestingly, ignoring the first part of the

TABLE V
COMPARISON OF ACTION FORECASTING METHODS USING BREAKFAST DATASET ONLY USING FEATURES. THE BEST RESULTS USING FISHER VECTOR (FV) FEATURES ARE UNDERLINED. OVERALL BEST RESULTS ARE SHOWN IN BOLD. ALL RESULTS FROM [3] AND [21] USE FRAME LEVEL ANNOTATIONS AND THEREFORE FULLY SUPERVISED. CASES WHERE OUR WEAKLY SUPERVISED METHOD OUTPERFORMS PRIOR SUPERVISED STATE-OF-THE-ART ARE UNDERLINED WITH RED COLOUR.

observation (%)	20%				30%			
prediction (%)	10%	20%	30%	50%	10%	20%	30%	50%
Grammar[3]	16.60	14.95	13.47	13.42	21.10	18.18	17.46	16.30
Nearest Neighbor[3]	16.42	15.01	14.47	13.29	19.88	18.64	17.97	16.57
RNN[3]	18.11	17.20	15.94	15.81	21.64	20.02	19.73	19.21
CNN[3]	17.90	16.35	15.37	14.54	22.44	20.12	19.69	18.76
Time-Condition [21]	18.41	17.21	16.42	15.84	22.75	20.44	19.64	19.75
Our fully supervised - FV	18.75	18.35	17.78	17.00	23.98	21.93	19.70	19.58
Our fully supervised - I3D	**23.03**	**22.28**	**22.00**	**20.85**	**26.50**	**25.00**	**24.08**	**23.61**
Our weakly supervised - FV	18.60	16.73	14.80	14.65	23.80	21.15	19.25	16.83
Our weakly supervised - I3D	21.70	18.85	16.65	14.58	26.20	21.78	20.43	16.50

TABLE VI
COMPARISON OF ACTION FORECASTING METHODS USING 50SALADS DATASET ONLY USING FEATURES. WE REPORT RESULTS USING FISHER VECTOR (FV) FEATURES USED IN PREVIOUS METHODS. OVERALL BEST RESULTS ARE SHOWN IN BOLD. WHEN OUR WEAKLY SUPERVISED METHOD OUTPERFORMS PRIOR METHODS, IT IS UNDERLINED. ALL RESULTS FROM [3] AND [21] USE FRAME LEVEL ANNOTATIONS AND THEREFORE FULLY SUPERVISED.

observation (%)	20%				30%			
prediction (%)	10%	20%	30%	50%	10%	20%	30%	50%
Grammar[3]	24.73	22.34	19.76	12.74	29.65	19.18	15.17	13.14
Nearest Neighbor[3]	19.04	16.1	14.13	10.37	21.63	15.48	13.47	13.90
RNN[3]	30.06	25.43	18.74	13.49	30.77	17.19	14.79	9.77
CNN[3]	21.24	19.03	15.98	9.87	29.14	20.14	17.46	10.86
Time-Condition [21]	32.51	27.61	21.26	15.99	35.12	27.05	22.05	15.59
Our fully supervised - FV	**39.32**	**31.39**	**27.01**	**23.88**	**41.73**	**32.73**	**31.44**	**26.39**
Our weakly supervised - FV	36.41	26.33	23.40	15.45	35.38	26.37	23.74	18.44

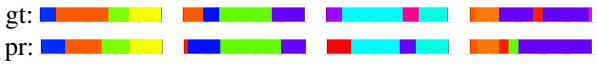

Fig. 5. Illustration of ground truth and forecasted actions for some random videos. Each color represents an action.

loss term also leads to poor performance. This shows that the models' ability to obtain a good representation for observed and unobserved future frames is important when forecasting future actions. This also somewhat confirms our hypothesis on mental time travel discussed in the introduction. Specifically, we assume that humans correlate prior experiences and examples with the current scenario to perform mental time travel. In this regard, it seems a better understanding of the past events perhaps help to improve future predictions.

Visual illustration of some predictions are shown in figure 5. Interestingly, most of the time our method is able to get the action class correctly, although the temporal extent is not precise. Furthermore, there is significant smoothness in the prediction that we believe is due to the sequential learning used in our method.

Similarly, now we provide results on 50Salads dataset using the five fold cross validation with provided splits in [43] which is the protocol used in [3], [21]. 50Salads dataset has 50 videos with 17 fine-grained action classes where average length of a video is 6.4 minutes and contain 20 action instances per video on average. We report the accuracy of predicted frames as mean over classes (MoC) and use the same Fisher vector features used in prior methods [3]. Results are reported in Table VI. We see a significant improvement in results on this dataset compared to the improvements seen in the Breakfast dataset. The recent Time-Condition [21] method obtains an average improvement of 4.62% over prior RNN model of [3]. Interestingly, our fully supervised method obtains a significant average improvement of 7.09% over the Time-Condition [21] model. Our weakly supervised method is 6.05% lower than our supervised model, yet obtains better results than supervised Time-Condition [21] method by obtaining an average improvement of 1.04% . Compared to the breakfast dataset, the improvement in the 50Salads dataset is positively surprising. By visual inspection of the data, we also notice that there is high temporal correlation in 50 Salads dataset, which might positively influence our model to generate accurate pseudo representation for future. Once again, we attribute this improvement to the model architecture shown in figure 3, the effective use of attention mechanism, use of pseudo hidden states to represent future frame representation and effective use of new loss functions.

VI. CONCLUSION.

In this paper we presented a method to predict future action sequence from a partial observation of a video using a GRU-based encoder-decoder machine translation technique. We showed the effectiveness of regularizing the cross-entropy loss for this task by catering the uncertainty of future predictions and the proposed optimal transport loss allowed us to further improve results. We observed that conditioning on few past video frames is not sufficient to forecast future

actions accurately. It is better to make use of all available information and use attention mechanism to select the most relevant frames in the partially observed video. The attention mechanism helped our model to better exploit the context of activity and obtain accurate future predictions.

Weakly supervised action forecasting is an important problem and in this work we proposed an effective method by taking advantage of an architecture that is designed to correlate observed feature sequence with the future action sequence. Our weakly supervised action forecasting model used an GRU encoder, three dedicated decoders and used an effective attention mechanism to obtain accurate actions for the future. Using this attention method, our model predicted labels for future unseen frames at test time without using frame specific action labels during training. It obtained competitive results compared to prior fully supervised methods and sometimes even outperformed them. Our method is conceptually simple and potentially useful for many practical applications where one can train with easily obtainable coarse annotations of videos. We believe our findings are insightful and useful for the development of future action forecasting methods.

ACKNOWLEDGMENT

This research/project is supported by the National Research Foundation, Singapore under its AI Singapore Programme (AISG Award No: AISG-RP-2019-010). Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not reflect the views of National Research Foundation, Singapore

REFERENCES

[1] T. Suddendorf and M. C. Corballis, "The evolution of foresight: What is mental time travel, and is it unique to humans?" *Behavioral and brain sciences*, vol. 30, no. 3, pp. 299–313, 2007.

[2] K. M. Kitani, B. D. Ziebart, J. A. Bagnell, and M. Hebert, "Activity forecasting," in *ECCV*. Springer, 2012, pp. 201–214.

[3] Y. Abu Farha, A. Richard, and J. Gall, "When will you do what?-anticipating temporal occurrences of activities," in *CVPR*, 2018, pp. 5343–5352.

[4] T. Lan, T.-C. Chen, and S. Savarese, "A hierarchical representation for future action prediction," in *ECCV*. Springer, 2014, pp. 689–704.

[5] K.-H. Zeng, S.-H. Chou, F.-H. Chan, J. Carlos Niebles, and M. Sun, "Agent-centric risk assessment: Accident anticipation and risky region localization," in *CVPR*, 2017, pp. 2222–2230.

[6] Y. Kong, S. Gao, B. Sun, and Y. Fu, "Action prediction from videos via memorizing hard-to-predict samples," in *AAAI*, 2018.

[7] M. Sadegh Aliakbarian, F. Sadat Saleh, M. Salzmann, B. Fernando, L. Petersson, and L. Andersson, "Encouraging lstms to anticipate actions very early," in *ICCV*, 2017, pp. 280–289.

[8] H. S. Koppula and A. Saxena, "Anticipating human activities using object affordances for reactive robotic response," *TPAMI*, vol. 38, no. 1, pp. 14–29, 2015.

[9] H. Gammulle, S. Denman, S. Sridharan, and C. Fookes, "Forecasting future action sequences with neural memory networks," in *BMVC*, 2019.

[10] T. Mahmud, M. Hasan, and A. K. Roy-Chowdhury, "Joint prediction of activity labels and starting times in untrimmed videos," in *ICCV*, 2017, pp. 5773–5782.

[11] J. Zhang, J. Du, S. Zhang, D. Liu, Y. Hu, J. Hu, S. Wei, and L. Dai, "Watch, attend and parse: An end-to-end neural network based approach to handwritten mathematical expression recognition," *Pattern Recognition*, vol. 71, pp. 196–206, 2017.

[12] M. S. Ryoo, "Human activity prediction: Early recognition of ongoing activities from streaming videos," in *ICCV*. IEEE, 2011, pp. 1036–1043.

[13] Y. Kong, D. Kit, and Y. Fu, "A discriminative model with multiple temporal scales for action prediction," in *ECCV*. Springer, 2014, pp. 596–611.

[14] Y. Shi, B. Fernando, and R. Hartley, "Action anticipation with rbf kernelized feature mapping rnn," in *ECCV*, 2018, pp. 301–317.

[15] C. Vondrick, H. Pirsiavash, and A. Torralba, "Anticipating visual representations from unlabeled video," in *CVPR*, 2016, pp. 98–106.

[16] A. Jain, A. Singh, H. S. Koppula, S. Soh, and A. Saxena, "Recurrent neural networks for driver activity anticipation via sensory-fusion architecture," in *ICRA*. IEEE, 2016, pp. 3118–3125.

[17] S. Ma, L. Sigal, and S. Sclaroff, "Learning activity progression in lstms for activity detection and early detection," in *CVPR*, 2016, pp. 1942–1950.

[18] K.-H. Zeng, W. B. Shen, D.-A. Huang, M. Sun, and J. Carlos Niebles, "Visual forecasting by imitating dynamics in natural sequences," in *ICCV*, 2017, pp. 2999–3008.

[19] Y. Wang, M. Long, J. Wang, Z. Gao, and S. Y. Philip, "Predrnn: Recurrent neural networks for predictive learning using spatiotemporal lstms," in *NIPS*, 2017, pp. 879–888.

[20] C. Rodriguez, B. Fernando, and H. Li, "Action anticipation by predicting future dynamic images," in *ECCV*, 2018, pp. 0–0.

[21] Q. Ke, M. Fritz, and B. Schiele, "Time-conditioned action anticipation in one shot," in *CVPR*, 2019, pp. 9925–9934.

[22] S. Qi, S. Huang, P. Wei, and S.-C. Zhu, "Predicting human activities using stochastic grammar," in *ICCV*, 2017, pp. 1164–1172.

[23] N. Rhinehart and K. M. Kitani, "First-person activity forecasting with online inverse reinforcement learning," in *ICCV*, 2017, pp. 3696–3705.

[24] S. Z. Bokhari and K. M. Kitani, "Long-term activity forecasting using first-person vision," in *ACCV*. Springer, 2016, pp. 346–360.

[25] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *NIPS*, 2014, pp. 3104–3112.

[26] P. Nguyen, T. Liu, G. Prasad, and B. Han, "Weakly supervised action localization by sparse temporal pooling network," in *CVPR*, June 2018.

[27] B. Fernando, C. Tan, and H. Bilen, "Weakly supervised gaussian networks for action detection," in *WACV*, 2020, pp. 537–546.

[28] D.-A. Huang, L. Fei-Fei, and J. C. Niebles, "Connectionist temporal modeling for weakly supervised action labeling," in *ECCV*. Springer, 2016, pp. 137–153.

[29] K. K. Singh and Y. J. Lee, "Hide-and-seek: Forcing a network to be meticulous for weakly-supervised object and action localization," in *ICCV*, 2017.

[30] Z. Shou, H. Gao, L. Zhang, K. Miyazawa, and S.-F. Chang, "Autoloc: Weaklysupervised temporal action localization in untrimmed videos," in *ECCV*, 2018, pp. 162–179.

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

[32] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *arXiv preprint arXiv:1409.0473*, 2014.

[33] S. Venugopalan, M. Rohrbach, J. Donahue, R. Mooney, T. Darrell, and K. Saenko, "Sequence to sequence - video to text," in *ICCV*, December 2015.

[34] H. Yu, J. Wang, Z. Huang, Y. Yang, and W. Xu, "Video paragraph captioning using hierarchical recurrent neural networks," in *CVPR*, 2016, pp. 4584–4593.

[35] N. Srivastava, E. Mansimov, and R. Salakhudinov, "Unsupervised learning of video representations using lstms," in *ICML*, 2015, pp. 843–852.

[36] G. Peyr and M. Cuturi, "Computational optimal transport," *Foundations and Trends in Machine Learning*, vol. 11 (5-6), pp. 355–602, 2019. [Online]. Available: https://arxiv.org/abs/1803.00567

[37] J. Feydy, T. Séjourné, F.-X. Vialard, S.-i. Amari, A. Trouve, and G. Peyré, "Interpolating between optimal transport and mmd using sinkhorn divergences," in *AIStat*, 2019, pp. 2681–2690.

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

[39] M. Rohrbach, S. Amin, M. Andriluka, and B. Schiele, "A database for fine grained activity detection of cooking activities," in *CVPR*. IEEE, 2012, pp. 1194–1201.

[40] H. Kuehne, A. Arslan, and T. Serre, "The language of actions: Recovering the syntax and semantics of goal-directed human activities," in *CVPR*, June 2014.

[41] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu, "Bleu: a method for automatic evaluation of machine translation," in *ACL*. Association for Computational Linguistics, 2002, pp. 311–318.

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

[43] S. Stein and S. J. McKenna, "Combining embedded accelerometers with computer vision for recognizing food preparation activities," in *Proceedings of the ACM international joint conference on pervasive and ubiquitous computing*, 2013, pp. 729–738.