Future Moment Assessment for Action Query

Qiuhong Ke[1], Mario Fritz[2], Bernt Schiele[3]
[1]The University of Melbourne, Melbourne, Australia
[2]CISPA Helmholtz Center for Information Security, Saarbrücken, Germany
[3]Max Planck Institute for Informatics, Saarbrücken, Germany
qiuhong.ke@unimelb.edu.au fritz@cispa.saarland schiele@mpi-inf.mpg.de

Abstract

In this paper, we aim to tackle the task of Assessing Future Moment of an Action of Interest (AFM-AI). The goal of this task is to assess if an action of interest will happen or not as well as the starting moment of the action. We aim to assess starting moments at any time-horizon of the future. To this end, we tackle the regression task of the starting moments as a generation task using a Deterministic Residual Guided Variational Regression Module (DR-VRM), which is built on a Variational Regression Module (VRM) and a deterministic residual network. The VRM takes the uncertainty into account and is capable of generating diverse predictions for the starting moment. The deterministic network encourages the VRM to learn from deterministic residual information in order to generate more precise predictions for moment assessment. Experimental results on three datasets clearly show that the proposed method is capable of generating both diverse and precise predictions of starting moments for query actions.

1. Introduction

While strong progress has been made on action and activity recognition [2, 17, 26, 35, 37, 41], many real-world applications demand to assess if and when a particular action of interest will occur. For example, in visual surveillance and self-driving cars, the systems need to infer dangerous actions to generate early alerts. In medical care systems, the system needs to assess fall of the elderly to prevent the accident. In human-robot interaction, the robot needs to assess if the human will conduct actions that require its assistance or interaction, thus to provide timely response. In these cases, there is no need to anticipate all the future actions after the observation. Instead, the system only needs to know if the action of interest will happen or not, and if so, when it will happen. We refer to this task as Assessing Future Moment of an Action of Interest (AFM-AI).

This task is mostly related to action anticipation, which aims to use the observation to predict the next action or future actions in a long-term horizon [6, 8, 19, 30, 32, 43]. To solve the AFM-AI task, one can use action anticipation methods to predict the future action sequence given the observation, and then use the predicted future sequence to verify whether the action of interest occurs or not, as well as the starting moment of the action of interest. While the action anticipation methods can be adapted for AFM-AI, the performance is somewhat unsatisfactory as anticipating actions for all future frames correctly is very challenging, not to mention its inefficiency due to intermediate anticipations.

In this paper, instead of solving the task of AFM-AI in such an indirect manner, we introduce a new framework to solve this problem. The proposed method treats the action of interest as query. It takes both the observation and the query action as inputs and directly outputs the assessment of whether the query action will happen or not as well as when it will happen. Compared to previous action anticipation methods, the proposed method does not need to anticipate all the other intermediate actions, which avoids intermediate anticipation errors and results in better performance.

More specifically, we address arbitrary actions of interest that occur at any time-horizon after the observation. For some actions of interest, especially the long-term ones, the starting moment in the future is uncertain due to human activities that do not follow a fixed action order and vary in length. For example, to make a salad, one might first put tomatoes, cheese and then cucumbers after taking a bread, while the other one might first put cheese, cucumbers and then tomatoes. In this case, it is necessary to capture the uncertainty of the data for assessment. There are also some cases that the action of interest is highly related to the observation and the starting moment is less uncertain, which can be assessed in a deterministic manner. For example, after opening the trash can, the next action is to dispose the trash. In our model, we take both aspects of the data into account, and introduce a Deterministic Residual Guided Variational Regression Module (DR-VRM) for starting moment assessment. DR-VRM is built on a Variational Regression

Module (VRM) and a deterministic network. The VRM is inspired by the Conditional Variational Auto-Encoders (CVAE) [38]. In particular, we introduce a Temporal Reconstruction Loss to equip CVAE. The Temporal Reconstruction Loss prevents the network from over-fitting on the long-term starting moments and makes the network capable of capturing the uncertainty of the data, leading to diverse predictions for the starting moment. The deterministic network encourages the VRM to learn from deterministic residual signal to allow for more precise predictions for moment assessment.

The contributions of this paper are summarized as follows: 1) We investigate a new task of assessing future moment of an arbitrary action of interest that occurs at any time-horizon after the observation, which is very important for real-world applications. 2) We introduce a new DR-VRM, which allows for diverse and precise predictions of the starting moments. 3) We conduct extensive experiments and clearly show the advantages of the proposed method.

2. Related Works

Action Anticipation. Action anticipation aims to predict future unseen actions after the observation. Many efforts have been made for early action recognition [1,4,14–16,18, 22,23,25,27,34]. Recently, there is an increasing number of papers for action anticipation [6,8,9,12,19,29,30,32,42,43]. In [9], a new method is introduced to anticipate egocentric actions. The method contains two LSTMs to summarize the historical information and perform predictions of the future action. A novel method based on attention is also introduced to combine information of different modalities. In [8], a RNN and a CNN model are introduced to anticipate long-term action sequences. In [19], a time-conditioned method is introduced to anticipate long-term future actions.
Time to Event Prediction. In [31], several loss functions are introduced to predict if and when the shooting will happen in a basketball dataset, and to predict if and when a car will stop in a car driving dataset. The prediction horizon is limited to 10s. Our method is capable of handing arbitrary actions of interest and the staring moments of the actions can be at any time-horizon of the future.
Offline Action Detection. In this task, the entire video is observed and the goal is to detect the start and end time of each action. In [5], a Temporal Context Network, which contains a pair-wise sampling layer to capture the local context, is introduced to perform temporal localization. In [3], a TAL-Net is proposed to generate proposal and classify actions. In [11], a Cross-modal Temporal Regression Localizer is introduced for action localization via language query.
Early Action Detection. This task aims to detect an action as early as possible from a partial observation of the action. In [14], a Max-Margin Early Event Detectors built based on Structured Output SVM is introduced to recognize partial events. In [28], a new ranking loss is introduced to encourage LSTM to generate non-decreasing detection score for correct class with more observation.

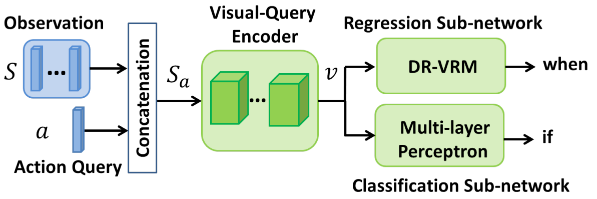

Figure 1: Architecture of the proposed method. The observed action sequence S and query action a are concatenated and passed to a Visual-Query Encoder. The generated feature v is then fed to two sub-networks to perform binary classification and regression. The Classification Sub-network is a Multi-layer Perceptron, which aims to address *if* the query action will happen or not in the future. The Regression Sub-network is a Deterministic Residual Guided Variational Regression Module (DR-VRM), which aims to address *when* the query action will happen.

Online Action Detection. The goal of this task is to recognize the action of the current frame and generate frame-level classification in an online manner, i.e., the future frames are not considered for the classification. In [45], a Temporal Recurrent Network is introduced to jointly perform action detection and anticipation. In [36], a network built on GAN [13] is introduced to generate hard negative samples for online detection of action start.
Variational Autoencoders. Variational learning allows models to learn with Gaussian latent variables [20, 21, 33]. Variational Auto-encoders (VAE) [21] has been used in [30] for action anticipation. Conditional variational auto-encoders (CVAE) [38] is built based on VAE [21] to models conditional distributions to predict diverse outputs. CVAE has been used for image description [44], future frame synthesis [46] and image colorization [7].

3. Assess Future Moment for Action Query

The overall architecture of the proposed method is shown in Figure 1. It takes both the observed sequence S and query action a as inputs, and generates two outputs: 1) a classification score to address *if* the query action will happen after the observation, and 2) a regression value to address *when* the query action will happen.

More specifically, given an observed action sequence $S \in \mathbb{R}^{c \times m}$ (c denotes the number of action classes and m denotes the length of the observation), the query action $a \in \mathbb{R}^{c \times 1}$ is concatenated with the sequence at each time-step. a is repeated for m time-steps before concatenation. The concatenation $S_a \in \mathbb{R}^{2c \times m}$ is fed to the Visual-Query Encoder to model joint information of the observation and the query action. The Visual-Query Encoder consists of several dilated temporal convolution layers, followed by a max-pooling layer to output a feature vector $v \in \mathbb{R}^{d\times 1}$ (d represents the dimension of the feature vector). v is passed to: 1) a Classification Sub-network, which outputs the probability that the query action will happen in the future, and 2) a Regression Sub-network, which outputs the starting moment of the query action.

Intuitively, the future is stochastic and uncertain. Given a past observed action sequence, there could be multiple possible future sequences. In other words, there is uncertainty in the answers of whether and when a query action occurs. It is thus necessary and important to generate diverse predictions to capture the multi-modal posterior over the output space. Below we describe the Classification Sub-network and Regression Sub-network, which are specifically designed to tackle this problem.

3.1. Classification Sub-network

As shown in Figure 1, the Classification Sub-network takes v as the input, and performs binary classification to output a probability score, which indicates the probability of the occurrence of the query action.

More specifically, the Classification Sub-network is a Multi-layer Perceptron, which consists of three fully-connected (FC) layers followed by a Sigmoid layer. To allow for diverse predictions in order to capture the multi-modal posterior over the output space, we use Monte Carlo dropout (MC-Dropout) [10] to equip the network. MC-Dropout is shown to be an approximation of the Bayesian interpretation. More specifically, we insert Dropout [39] layers before all the hidden layers. The Dropout layers are used during training and testing for diverse anticipation. The objective for the Classification Sub-network is to minimize the binary cross-entropy between the predictions and the data labels, which is formulated as:

$$\mathcal{L}_c = -\mathbb{E}\left[y_c \log(o_c) + (1-y_c)\log(1-o_c))\right] \tag{1}$$

where o_c denotes the output of the Classification Sub-network and y_c denotes the data label. $\mathbb{E}[\cdot]$ denotes statistical expectation.

3.2. Regression Sub-network

The Regression Sub-network takes v as the input and performs regression to output a value, which indicates the starting moment of the query action in the future.

Ideally, we can design a network similar to the Classification Sub-network and use MC-Dropout [10] to capture the uncertainty of the data for regression. However, the starting moment can be any positive value, rather than 0 or 1 in the binary classification task. This makes the distribution of starting moment more complex and the regression task more challenging. To effectively capture the data distribution and generate more precise and diverse predictions, we introduce a new method to assess the starting moment of the query action.

More specifically, we first introduce a Variational Regression Module (VRM), which aims to capture the uncertainty of the data and can generate diverse predictions for the starting moment. Based on VRM, we further introduce a Deterministic Residual Guided VRM (DR-VRM), which aims to generate both diverse and precise predictions. VRM and DR-VRM are inspired by CVAE [38]. Below we introduce the details.

3.2.1 Preliminary: Conditional Variational Auto-encoders

CVAE is a conditional generative model with Gaussian latent variables [38]. The idea of CVAE is to use the latent variable to model different modes of the distribution between the input and the output variables, thus to allow the network to learn one-to-many mapping and generate diverse predictions. Specifically, given an input variable x, CVAE aims to generate an output variable y from the distribution $p(y|x,z)$, where z is a latent variable. CVAE is trained to maximize the conditional log-likelihood. However the objective function is intractable. A surrogate objective function is the variational lower bound of the log-likelihood (derivation of the variational lower bound can be found in [38]):

$$\begin{aligned}\log p(y|x) \geq &- D_{KL}\left(q(z|x,y) \parallel p(z|x)\right) \\ &+ \mathbb{E}_q[\log p(y|x,z)]\end{aligned} \tag{2}$$

where D_{KL} denotes the KL divergence of two distributions. $q(z|x,y)$ denotes the Gaussian distribution output by the encoder of CVAE (which is also referred to as the recognition network). Specifically, the encoder takes both x and y as the inputs and generates mean and variance of a Gaussian distribution where z is sampled from. $p(z|x)$ is the prior distribution of the latent variable, which is set to standard normal distribution. $p(y|x,z)$ is the output of the decoder of the CVAE, i.e., the decoder takes x and z as the inputs and outputs y. The first part of the objective minimizes the KL divergence between the distribution output by the encoder and the prior distribution. The second part maximizes the data log-likelihood conditioned on and x and z, where z is sampled from the distribution output by the encoder.

3.2.2 Variational Regression Module

The main idea of VRM is to tackle the regression task of the starting moment as a variational generation task conditioned on the input information. Basically, VRM is built based on CVAE [38]. The overall architecture of VRM is

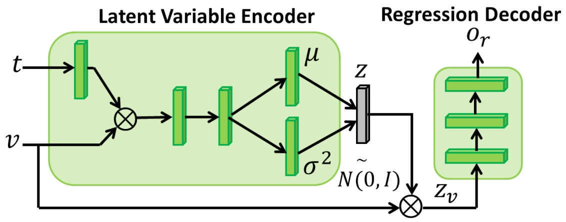

Figure 2: Architecture of Variational Regression Module (VRM). v is the output of the Visual-Query Encoder (Figure 1). t is the ground-truth starting moment of the query action. $\otimes$ denotes element-wise multiplication. μ and σ^2 are mean and variance of the Gaussian distribution where the latent variable z is sampled. o_r is the predicted starting moment.

shown in Figure 2. In the case, the input variable of the CVAE is the output of the Visual-Query Encoder (shown in Figure 1) v, and the output variable is the ground-truth starting moment of the query action t. We use only the positive samples (i.e., the query action happens in the future) for training. For the negative samples, t is set to a random number (e.g., 0) and the regression loss is not calculated during training, i.e., the negative samples are not involved in training the regression model. Using only the positive samples to train the regression model ensures that the regression model is learned from the true distribution of the starting moments. t is fed to an embedding layer and then merged with v using element-wise multiplication. One can also use concatenation. The outputs of the encoder are the mean μ and variance σ^2 of the Gaussian distribution, where z is sampled from during training. z is merged with v and the combination z_v is fed to a regression decoder to generate o_r, which is the prediction of the starting moment. o_r is a scalar, i.e., each time the network takes one query action as the input and generates one number as the prediction of the starting moment.

The objective of VRM is the variational lower bound shown in Equation 2. In particular, the input variable x in Equation 2 is v and the output variable y in Equation 2 is t. Specifically, the data log-likelihood of VRM is formulated as:

$$\log p(t|v,z) = -\frac{(t-o_r)^2}{2\sigma^2} - \frac{1}{2}\log(2\pi\sigma^2) \tag{3}$$

Minimizing mean square error is equivalent to maximize the log-likelihood, except that the variance σ^2 is ignored. In our task, we observe that the variance of the starting moments of long-term query actions are larger than those of the short-term ones. This could be partially due to the fact that the observation and long-term query actions are not closely related, and there could be multiple possible intermediate actions between them. This makes the starting moment of the long-term query action more stochastic. We therefore set the variance of each sample as a function of the starting moment in order to follow the trend of the data, i.e., smaller variance for the shorter-term query actions while larger variance for the longer-term ones. More specifically, we set $\sigma^2 = t^\tau$, where τ is a hyper-parameter.

During training, the loss function of VRM is the negative form of variational lower bound, i.e., the combination of the negative data log-likelihood and the KL divergence between the distribution output by the latent variable encoder and the prior distribution. Formally,

$$\mathcal{L}_r = \ell_r + \mathbb{E}\left[D_{KL}\left(q(z|t,v) \parallel p(z|v)\right)\right] \tag{4}$$

$$\ell_r = \mathbb{E}\left[\frac{(t-o_r)^2}{2t^\tau} + \frac{1}{2}\log(2\pi t^\tau)\right] \tag{5}$$

ℓ_r is referred to as the Temporal Reconstruction Loss, which produces less penalty for the long-term query actions than the short-term ones under the same error term $t - o_r$. This prevents the network from over-fitting and enables the network converge easily.

Once the network is trained, the encoder is removed. During testing, z is sampled from the prior distribution as the approximation, which is then combined with v to assess the starting moment using the regression decoder. z can be sampled multiple times to generate diverse predictions.

3.2.3 Deterministic Residual Guided Variational Regression Module

Intuitively, if the query action is highly relevant to the observation, e.g., next action after the observation, the starting moment of the query action can be inferred using a deterministic manner. In this case, the diverse predictions generated by VRM are not precise. In this section, we introduce a DR-VRM to achieve diverse and precise regressions. Basically, DR-VRM consists of a deterministic network and VRM. The main idea of DR-VRM is to guide the learning of the VRM using the deterministic network. More specifically, VRM is encouraged to learn the residual information that cannot be captured by the deterministic network. The residual is zero if the deterministic network is optimal. This enables the network to generate more precise predictions.

The overall architecture of DR-VRM is shown in Figure 3. The inputs v and t and the latent variable encoder of DR-VRM are the same as that of VRM. The regression decoder consists of two streams. One stream takes the conditional latent variable z_0 as input. This stream combined with the latent variable encoder has the same architecture of VRM. This stream is referred to as the variational stream as it is capable of generating diverse predictions. Another stream takes v as input, which can only generate point estimates and it is referred to as the deterministic stream. These two streams consist of the same architecture. The output of the deterministic stream at each layer is passed to the corresponding layer of the variational stream to form a new out-

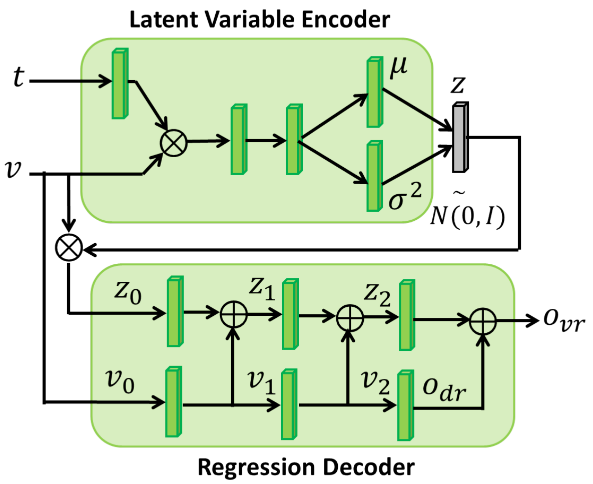

Figure 3: Architecture of Deterministic Residual Guided Variational Regression Module (DR-VRM). v is the output of the Visual-Query Encoder (Figure 1). t is the ground-truth starting moment of the query action. $\bigotimes$ and $\bigoplus$ denote element-wise multiplication and addition, respectively. μ and σ^2 are mean and variance of the Gaussian distribution where the latent variable z is sampled. o_{vr} is the predicted starting moment.

put of the corresponding layer of the variational stream. We denote the functional representation of the i^{th} FC layer of the variational stream and the deterministic stream as $f_i(\cdot)$ and $g_i(\cdot)$, respectively. As shown in Figure 2, the output of the i^{th} fully-connected (FC) layer of the variational stream z_i is formulated as: $z_i = f_i(z_{i-1}) \oplus v_i$, where $v_i = g_i(v_{i-1})$ denotes the output of the $(i)^{th}$ FC layer of the deterministic stream ($v_0 = v$). $\oplus$ denotes element-wise addition. The final output of the variational stream is formulated as $o_{vr} = f_3(z_2) \oplus o_{dr}$. As shown in Figure 2, $o_{dr} = g_3(v_2)$ is the output of the deterministic stream. o_{vr} is a scalar, which denotes the starting moment of a query action.

During training, we optimize both the variational stream and the deterministic stream. The cost function of the variation stream is the same as VRM, which is the combination of the data negative log-likelihood and the KL divergence between the encoding distribution and the prior, as shown in Equation 4 and Equation 5. The cost function of the deterministic stream is the data negative log-likelihood, as shown in Equation 5 (in this case, $o_r = o_{dr}$). During testing, similar to VRM, the latent variable encoder is removed and z is sampled from prior standard normal distribution. The output of the variational stream o_{vr} is used as the final prediction for evaluation.

3.3. Implementation Details

The Visual-Query Encoder contains three dilated temporal convolutional layers with kernal size 3, and dilated rate 2, 4 and 8, respectively. The sizes of filters are set to 32, 64, 128, respectively. The Classification Sub-network is a Multi-layer Perceptron, which consists of three FC layers followed by a Sigmoid layer. The dropout rate is set to 0.5. The numbers of units in the hidden layers of the Classification Sub-network, the latent variable encoder and the regression decoder of VRM and DR-VRM are set to 256 and 512, respectively. In the latent variable encoders of VRM and DR-VRM, t is fed to a Multi-layer Perception with 64-unit hidden layer and 128-unit output layer. One can also use an embedding layer. The numbers of units of the output layer for mean and variance are set to 128. τ in Equation 5 is set to 1, which is selected by validation set. We use a sliding window to process each video to generate sub-sequences as observations for experiments. The length of observation is set to 60s. During training, we randomly sample an action that occurs in the future after the observation to form a positive pair. An action might occur multiple times after the observation. We use the first time it occurs as the starting moment. To generate negative pair, we randomly sample an action that does not occur in the future from the whole action set of the dataset. Both the observations and query actions are ground-truth action classes, i.e., one-hot class vectors. The objects and scenes can also be used to improve the prediction performance. We use Adam as the optimizer in all experiments. The learning rate is set to 0.001 and the batch size is set to 64.

4. Experiments

We conduct experiments with the following settings:
Deterministic Residual Guided Variational Query Network (DR-VQNet): This network takes both the observation and the query action as inputs for moment assessment, as shown in Figure 1. The Regression Sub-network is DR-VRM. During testing, the encoder of the DR-VRM is removed and z is randomly sampled from the standard normal distribution for multiple times. The random sampling process of z leads to diverse predictions for the same input.
Variational Query Network (VQNet): The architecture is similar to DR-VQNet, except that the Regression Sub-network is VRM. During testing, z is randomly sampled from the standard normal distribution for multiple times to generate diverse predictions.
The comparison methods include:
Deterministic Query Network (DQNet): The architecture is similar to DR-VQNet, except that the variational stream in the Regression Sub-network is removed and the Classification Sub-networks is a Multi-layer Perceptron.
Anticipation Network (AntiNet) [19]: Given an observed sequence, this method adapts the anticipation method [19] to anticipate the future action sequence. Then the future sequence is used to check if and when a query action happens.
CVAE [38]: This method is similar to VQNet except that

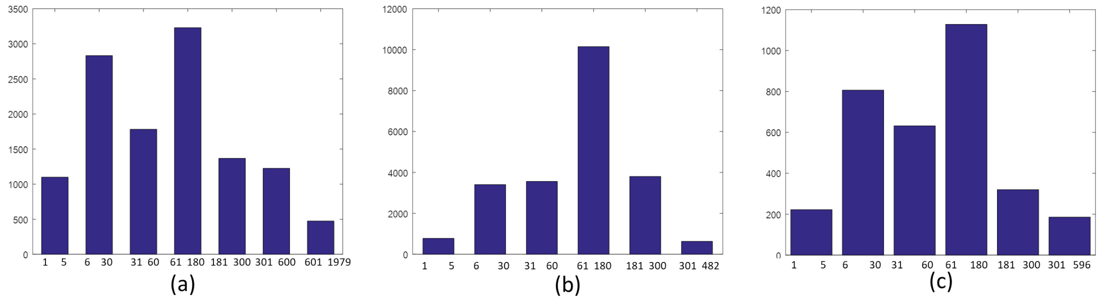

Figure 4: The statistical distribution of starting moments of query actions in (a) Epic-kitchen Dataset [6], (b) 50Salads Dataset [40] and (c) Breakfast Dataset [24]. The x axis represents the temporal interval (in seconds) of the starting moments (after the observations) of the query actions. For example, '61-180' represents that the starting moments of the query actions are between 61s and 180s after the observations. The y axis represents the number of samples.

the Regression Sub-network is the original CVAE [38] and the regression loss is mean square error rather than the Temporal Reconstruction Loss. Similar to VQNet, this method is capable of generating diverse predictions by randomly sampling z from the standard normal distribution.

Gaussian Mixture Model (GMM) [31]: This method is introduced in [31] to predict time of a future action and has achieved the best performance. We compare with this method to show the advantage of the proposed method. During testing, the diverse predictions can be generated by randomly sampling values from the output distribution.

MC-Dropout [10]: This method is similar to the proposed framework in Figure 1, except that the Regression and Classification Sub-networks have the same architecture, with MC-Dropout [10] layers inserted before each layer. The basic idea is to use dropout to approximate the Bayesian variational inference. Specifically, the dropout is enabled during training and testing, i.e., when the network processes an input, some connections in the layers are randomly disconnecting. Consequently, feeding the same input to the network for multiple times can result in diverse predictions.

4.1. Datasets

Epic-kitchen Dataset [6] is a large first-person video dataset. This dataset contains 125 classes, which are captured by 28 subjects. We follow [19] to randomly split the videos into 7 splits, and use 7-folder-cross-validation for evaluation. The videos have been segmented into 89600 video sequences using temporal sliding windows, which are used as observations. For each sequence, we query the most common 20 actions in the dataset. The statistical distribution of the starting moments of the query actions averaged across the splits is shown in Figure 4 (a).

50Salads Dataset [40] contains 50 videos of making salads performed by 25 subjects. There are 17 fine-grained action classes. This dataset has been split into 5 splits. The experiments are conducted using five training/testing splits and the results averaged across the five splits are reported. We use a temporal sliding window to segment the videos and generate around 15000 sequences, which are used observations. For each sequence, we query all the actions in the dataset. The statistical distribution of the starting moments (after the observations) of the query actions averaged across the splits is shown in Figure 4 (b).

Breakfast Dataset [24] contains 1712 videos, which are performed by 52 subjects preparing breakfast. There are 47 fine-grained action classes and a starting/ending action. This dataset has been split into 4 splits. We use the four training/testing splits for experiments, and report the results averaged across the four splits. Similar to the previous two datasets, we use a temporal sliding window to generate around 6500 sequences, which are as observations. For each sequence, we query the 47 actions in the dataset. The statistical distribution of the starting moments (after the observations) of the query actions is shown in Figure 4 (c).

These three datasets are about food preparation in the kitchen environment, which are popularly used for action anticipation. However, our method has no constraint of the query actions and can also be used in other scenarios.

4.2. Evaluation Metrics

To evaluate the performance of regression, we use root-mean-square error (**RMSE**) as the metric to measure the difference between the predicted and ground-truth starting moment (in second) for all the positive actions (actions that occur in the future). During testing, for the stochastic methods, we generate 100 predictions, and then calculate the mean of the predictions to evaluate RMSE. We also calculate the negative log-likelihood (**NLL**) to evaluate how well a model fits the data. A smaller value of NLL indicate a better model. The value can also be used to evaluate the uncertainty quality [10]. To evaluate the classification performance, we use the area under ROC curve (**AUC**) and the average per-class accuracy (**ACC**) as the metrics. For the

	Epic-kitchen Dataset				50Salads Dataset				Breakfast Dataset			
Method	RMSE ↓	NLL ↓	AUC ↑	ACC ↑	RMSE ↓	NLL ↓	AUC ↑	ACC ↑	RMSE ↓	NLL ↓	AUC ↑	ACC ↑
DQNet	202.88	14.87	0.75	67.96%	75.81	12.28	0.85	76.22%	78.16	11.94	0.98	92.70%
AntiNet [19]	217.67	17.15	-	54.14%	114.73	16.62	-	67.53%	100.34	13.61	-	86.54%
GMM [31]	184.05	5.78	0.75	67.64%	82.89	8.10	0.84	72.39%	102.75	8.38	**0.99**	**94.55%**
MC-Dropout [10]	205.23	13.09	0.73	66.76%	76.05	10.04	0.85	75.84%	76.20	10.19	0.98	92.72%
CVAE [38]	178.94	11.71	0.69	65.06%	84.75	3.91	0.69	63.55%	108.84	4.57	0.92	85.87%
VQNet (ours)	177.20	4.62	0.74	66.32%	86.08	3.75	**0.89**	**79.05%**	83.02	**3.64**	0.98	94.45%
DR-VQNet (ours)	**169.94**	**4.36**	**0.76**	**68.40%**	**67.51**	**3.69**	**0.89**	78.84%	**70.62**	3.71	0.98	94.04%

Table 1: Results of different methods. ↑ denotes higher value is better. ↓ denotes lower value is better.

	Epic-kitchen Dataset				50Salads Dataset				Breakfast Dataset			
t/s	RMSE ↓	NLL ↓	AUC ↑	ACC ↑	RMSE ↓	NLL ↓	AUC ↑	ACC ↑	RMSE ↓	NLL ↓	AUC ↑	ACC ↑
$t \leq 5$	66.99	3.32	**0.80**	**74.67%**	**35.05**	5.28	0.67	62.45%	36.38	4.56	0.95	90.24%
$5 < t \leq 30$	71.68	**2.97**	0.78	72.32%	42.82	3.65	0.73	67.48%	**35.93**	3.10	0.96	91.28%
$30 < t \leq 60$	65.81	3.30	0.70	64.67%	46.28	3.21	0.72	66.42%	36.47	**3.05**	0.96	91.81%
$60 < t \leq 180$	**59.72**	3.79	0.62	57.42%	45.51	**3.10**	0.76	69.58%	55.98	3.65	**0.97**	93.84%
$180 < t \leq 300$	130.35	4.69	0.54	50.10%	106.14	4.40	0.78	70.78%	107.53	4.64	**0.97**	93.65%
$t > 300$	415.40	9.38	0.49	44.16%	169.73	7.30	**0.84**	**73.43%**	180.13	5.72	0.96	**94.25%**

Table 2: Results of DR-VQNet over different time windows of the starting moment t.

methods with MC-Dropout, we compute the mean of the 100 predictions to evaluate the classification performance.

4.3. Results

Table 1 compares the results of different methods. The proposed VQNet and DR-VQNet achieve much better performance than the anticipation method (AntiNet [19]). In particular, the RMSE values of the AntiNet [19] for assessing starting moments of long-term query actions (starting moment $t > 300s$) are 536.02, 275.56 and 303.11 on the Epic-kitchen Dataset, 50Salads Dataset and Breakfast Dataset, respectively, which are much worse than the long-term anticipation results of the DR-VQNet as shown in the last row of Table 2. The AntiNet [19] assesses the future moment by anticipating the future sequence, which is time-consuming and challenging. The incorrect anticipated future sequence leads to the worse performance in moment assessment. Note that the AntiNet [19] does not directly perform binary classification, thus there is no AUC value for this method. Compared to the DQNet, the VQNet achieves a much better (smaller) NLL value. This means that the VQNet is capable of capturing the uncertainty of the data. However, the predictions of the VQNet is not precise, resulting in worse RMSE than the DQNet in the 50Salads Dataset and Breakfast Dataset. The DR-VQNet achieves a comparable NLL value with the VQNet, while the RMSE value of the DR-VQNet is much lower than the VQNet. The DR-VQNet incorporates a deterministic network to encourage the variational stream to learn from deterministic residual information. This enables the network to generate more precise predictions. The RMSE of the VQNet is smaller than the DQNet in the Epic-kitchen dataset as this dataset is more challenging and assessing the future moment with uncertainty is more beneficial. We have provided some qualitative examples in the supplementary material. Table 2 shows the performance of the DR-VQNet when the starting moment t is at different time windows. When $t > 300s$, the performance on the Epic-kitchen dataset is much worse than that on the other two datasets. As shown in Figure 4, the portion of $t > 300s$ and the maximum value of t of the Epic-kitchen dataset are much larger than those of the other two datasets. This makes it more challenging for long-term future moment assessment on the Epic-kitchen dataset.

From Table 1 we can also see that the NLL values of VQNet and DR-VQNet are much better (smaller) than MC-Dropout [10] and GMM [31]. Both the VQNet and DR-VQNet leverage the starting moment t to model the distribution during training, which can fit the data and capture the uncertainty better, resulting in better NLL values. The VQNet and DR-VQNet also outperform CVAE [30]. Compared to the CVAE [30], the VQNet and DR-VQNet uses Temporal Reconstruction Loss for regression. The Temporal Reconstruction Loss produces less penalty for the long-term query actions, which prevents the network from overfitting and results in better performance.

Analysis on τ. As shown in Equation 5, we set the variance σ^2 equal to t^τ, where t is the starting time of the query action and τ is the hyper-parameter which is set to 1. The

	Epic-kitchen Dataset				50Salads Dataset				Breakfast Dataset			
τ	RMSE $\downarrow$	NLL $\downarrow$	AUC $\uparrow$	ACC $\uparrow$	RMSE $\downarrow$	NLL $\downarrow$	AUC $\uparrow$	ACC $\uparrow$	RMSE $\downarrow$	NLL $\downarrow$	AUC $\uparrow$	ACC $\uparrow$
0	317.69	14.95	0.65	61.25%	79.17	4.80	0.76	68.63%	76.48	4.88	0.92	84.21%
0.5	213.19	5.33	0.68	63.67%	74.06	3.72	0.87	77.05%	81.38	3.74	0.97	92.42%
1	**169.94**	**4.36**	0.76	68.40%	**67.51**	**3.69**	0.89	78.84%	**70.62**	**3.71**	0.98	94.04%
2	213.77	16.31	**0.78**	**69.94%**	99.62	14.77	**0.90**	**80.06%**	94.33	13.74	**0.99**	**94.89%**

Table 3: Results of DR-VQNet with different values of τ in Equation 5.

	Epic-kitchen Dataset				50Salads Dataset				Breakfast Dataset			
DRL	RMSE $\downarrow$	NLL $\downarrow$	AUC $\uparrow$	ACC $\uparrow$	RMSE $\downarrow$	NLL $\downarrow$	AUC $\uparrow$	ACC $\uparrow$	RMSE $\downarrow$	NLL $\downarrow$	AUC $\uparrow$	ACC $\uparrow$
Hid	170.16	**4.35**	0.74	66.23%	70.64	3.81	**0.89**	78.75%	**70.10**	3.76	**0.98**	**94.37%**
Output	173.61	4.51	0.74	66.49%	74.59	4.24	0.87	76.59%	71.74	3.97	**0.98**	93.23%
Proposed	**169.94**	4.36	**0.76**	**68.40%**	**67.51**	**3.69**	**0.89**	**78.84%**	70.62	**3.71**	**0.98**	94.04%

Table 4: Results of DR-VQNet with different deterministic residual layers (DRL) in the regression decoder.

z	RMSE $\downarrow$	NLL $\downarrow$	AUC $\uparrow$	ACC $\uparrow$
prior	206.63	13.75	0.73	66.53%
Proposed	**169.94**	**4.36**	**0.76**	68.40%

Table 5: Results of DR-VQNet with different sampling methods of the latent variable z on the Epic-kitchen dataset.

weights	RMSE $\downarrow$	NLL $\downarrow$	AUC $\uparrow$	ACC $\uparrow$
0.01	176.48	4.67	0.74	66.69%
0.1	175.03	4.56	0.74	66.15%
0.5	173.78	4.50	0.74	66.04%
1	**169.94**	**4.36**	**0.76**	**68.40%**
5	178.17	6.28	0.74	66.61%

Table 6: Results of DR-VQNet with different weights of the deterministic residual information on Epic-kitchen dataset.

results on the other values of τ are shown in Table 3. Using a smaller τ, the variance is smaller and the network is over-fitting. In contrast, the network is under-fitting for the regression task with a larger τ. In this case, the network focuses on the classification task, resulting in better classification performance and worse regression performance.

Analysis on Deterministic Residual Layers. As shown in Figure 3, in the regression decoder of the DR-VRM, the residual module consists of hidden layers and output layer, i.e., the hidden features and output of the deterministic stream are passed to the variational stream. We have conducted the following two baselines that pass only the hidden features (**Hid**) and only the output (**Output**) of the deterministic stream to the variational stream. The results are shown in Table 4. Using only the hidden features or output leads to worse performance in most cases.

Analysis on the Latent Variable z. To show the benefit of sampling the latent variable z from the encoder during training, we have conducted the following baseline (**prior**): using the prior standard normal distribution to sample z in DR-VQNet. The results on the Epic-kitchen dataset are shown in Table 5. Sampling z from the prior distribution requires a large number of samples to accurately estimate the Monte Carlo log-likelihood, which is usually impossible and results in worse performance. Instead, the proposed method uses the encoder to perform importance sampling, which can reduce the sampling space and model the data distribution more accurately, leading to better performance.

Analysis on the weight of the deterministic information. In the DR-VQNet, the deterministic information are directly added to the variational information, i.e., the weight of the deterministic information is equal to 1. We have applied a weight to the deterministic information before adding to the variational information. The results are shown in Table 6. The performance is smaller with a larger or smaller weight, while setting the weight to 1 results in the best performance. It shows the equal importance of both the variational and deterministic information.

5. Conclusion

In this paper, we have addressed a new task of anticipating future moment of an arbitrary action of interest which occurs at any time-horizon after the observation. We have proposed a Variational Regression Module (VRM), which is capable of generating diverse predictions. Furthermore, we have proposed a Deterministic Residual Guided Variational Regression Module (DR-VRM), which incorporates a deterministic network into the VRM for both diverse and precise predictions. We have conducted extensive experiments and analysis to show the capacity of the proposed method for generating more precise and diverse predictions.

References

[1] Mohammad Sadegh Aliakbarian, Fatemehsadat Saleh, Mathieu Salzmann, Basura Fernando, Lars Petersson, and Lars Andersson. Encouraging lstms to anticipate actions very early. In *IEEE International Conference on Computer Vision*, 2017.

[2] Joao Carreira and Andrew Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6299–6308, 2017.

[3] Yu-Wei Chao, Sudheendra Vijayanarasimhan, Bryan Seybold, David A Ross, Jia Deng, and Rahul Sukthankar. Rethinking the faster r-cnn architecture for temporal action localization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1130–1139, 2018.

[4] Lei Chen, Jiwen Lu, Zhanjie Song, and Jie Zhou. Part-activated deep reinforcement learning for action prediction. In *European Conference on Computer Vision*, pages 421–436, 2018.

[5] Xiyang Dai, Bharat Singh, Guyue Zhang, Larry S Davis, and Yan Qiu Chen. Temporal context network for activity localization in videos. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5793–5802, 2017.

[6] Dima Damen, Hazel Doughty, Giovanni Maria Farinella, Sanja Fidler, Antonino Furnari, Evangelos Kazakos, Davide Moltisanti, Jonathan Munro, Toby Perrett, Will Price, et al. Scaling egocentric vision: The epic-kitchens dataset. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 720–736, 2018.

[7] Aditya Deshpande, Jiajun Lu, Mao-Chuang Yeh, Min Jin Chong, and David Forsyth. Learning diverse image colorization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6837–6845, 2017.

[8] Yazan Abu Farha, Alexander Richard, and Juergen Gall. When will you do what?-anticipating temporal occurrences of activities. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

[9] Antonino Furnari and Giovanni Maria Farinella. What would you expect? anticipating egocentric actions with rolling-unrolling lstms and modality attention. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 6252–6261, 2019.

[10] Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *International Conference on Machine Learning*, pages 1050–1059, 2016.

[11] Jiyang Gao, Chen Sun, Zhenheng Yang, and Ram Nevatia. Tall: Temporal activity localization via language query. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5267–5275, 2017.

[12] Jiyang Gao, Zhenheng Yang, and Ram Nevatia. Red: Reinforced encoder-decoder networks for action anticipation. In *British Machine Vision Conference*, 2017.

[13] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, pages 2672–2680, 2014.

[14] Minh Hoai and Fernando De la Torre. Max-margin early event detectors. *International Journal of Computer Vision*, 107(2):191–202, 2014.

[15] Jian-Fang Hu, Wei-Shi Zheng, Lianyang Ma, Gang Wang, and Jianhuang Lai. Real-time RGB-D activity prediction by soft regression. In *European Conference on Computer Vision*, pages 280–296, 2016.

[16] Jian-Fang Hu, Wei-Shi Zheng, Lianyang Ma, Gang Wang, Jian-Huang Lai, and Jianguo Zhang. Early action prediction by soft regression. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2018.

[17] Noureldien Hussein, Efstratios Gavves, and Arnold WM Smeulders. Timeception for complex action recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 254–263, 2019.

[18] Qiuhong Ke, Mohammed Bennamoun, Senjian An, Farid Boussaid, and Ferdous Sohel. Human interaction prediction using deep temporal features. In *European Conference on Computer Vision*, pages 403–414, 2016.

[19] Qiuhong Ke, Mario Fritz, and Bernt Schiele. Time-conditioned action anticipation in one shot. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9925–9934, 2019.

[20] Durk P Kingma, Shakir Mohamed, Danilo Jimenez Rezende, and Max Welling. Semi-supervised learning with deep generative models. In *Advances in neural information processing systems*, pages 3581–3589, 2014.

[21] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. In *International Conference on Learning Representations*, 2014.

[22] Yu Kong, Dmitry Kit, and Yun Fu. A discriminative model with multiple temporal scales for action prediction. In *European Conference on Computer Vision*, pages 596–611. 2014.

[23] Yu Kong, Zhiqiang Tao, and Yun Fu. Deep sequential context networks for action prediction. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1473–1481, 2017.

[24] H. Kuehne, A. B. Arslan, and T. Serre. The language of actions: Recovering the syntax and semantics of goal-directed human activities. In *Proceedings of Computer Vision and Pattern Recognition Conference (CVPR), year =*.

[25] Tian Lan, Tsung-Chuan Chen, and Silvio Savarese. A hierarchical representation for future action prediction. In *European Conference on Computer Vision*, pages 689–704, 2014.

[26] Colin Lea, Michael D Flynn, Rene Vidal, Austin Reiter, and Gregory D Hager. Temporal convolutional networks for action segmentation and detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 156–165, 2017.

[27] Wenbin Li and Mario Fritz. Recognition of ongoing complex activities by sequence prediction over a hierarchical label space. In *IEEE Winter Conference on Applications of Computer Vision*, pages 1–9, 2016.

[28] Shugao Ma, Leonid Sigal, and Stan Sclaroff. Learning activity progression in lstms for activity detection and early detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1942–1950, 2016.

[29] Tahmida Mahmud, Mahmudul Hasan, and Amit K Roy-Chowdhury. Joint prediction of activity labels and starting times in untrimmed videos. In *IEEE International Conference on Computer Vision*, pages 5784–5793, 2017.

[30] Nazanin Mehrasa, Akash Abdu Jyothi, Thibaut Durand, Jiawei He, Leonid Sigal, and Greg Mori. A variational autoencoder model for stochastic point processes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3165–3174, 2019.

[31] Lukas Neumann, Andrew Zisserman, and Andrea Vedaldi. Future event prediction: If and when. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 1–9, 2019.

[32] Siyuan Qi, Siyuan Huang, Ping Wei, and Song-Chun Zhu. Predicting human activities using stochastic grammar. In *IEEE International Conference on Computer Vision*, pages 1164–1172, 2017.

[33] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *ICML*, 2014.

[34] MS Ryoo. Human activity prediction: Early recognition of ongoing activities from streaming videos. In *IEEE International Conference on Computer Vision*, pages 1036–1043, 2011.

[35] Xingjian Shi, Zhourong Chen, Hao Wang, Dit-Yan Yeung, Wai-Kin Wong, and Wang-chun Woo. Convolutional lstm network: A machine learning approach for precipitation nowcasting. In *Advances in Neural Information Processing Systems*, pages 802–810, 2015.

[36] Zheng Shou, Junting Pan, Jonathan Chan, Kazuyuki Miyazawa, Hassan Mansour, Anthony Vetro, Xavier Giro-i Nieto, and Shih-Fu Chang. Online detection of action start in untrimmed, streaming videos. In *European Conference on Computer Vision*, pages 534–551, 2018.

[37] Karen Simonyan and Andrew Zisserman. Two-stream convolutional networks for action recognition in videos. In *Advances in Neural Information Processing Systems*, pages 568–576, 2014.

[38] Kihyuk Sohn, Honglak Lee, and Xinchen Yan. Learning structured output representation using deep conditional generative models. In *Advances in Neural Information Processing Systems*, pages 3483–3491, 2015.

[39] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

[40] Sebastian Stein and Stephen J McKenna. Combining embedded accelerometers with computer vision for recognizing food preparation activities. In *Proceedings of the 2013 ACM international joint conference on Pervasive and ubiquitous computing*, pages 729–738, 2013.

[41] Du Tran, Lubomir Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. Learning spatiotemporal features with 3d convolutional networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4489–4497, 2015.

[42] Nam N Vo and Aaron F Bobick. From stochastic grammar to bayes network: Probabilistic parsing of complex activity. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2641–2648, 2014.

[43] Carl Vondrick, Hamed Pirsiavash, and Antonio Torralba. Anticipating visual representations from unlabeled video. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 98–106, 2016.

[44] Liwei Wang, Alexander Schwing, and Svetlana Lazebnik. Diverse and accurate image description using a variational auto-encoder with an additive gaussian encoding space. In *Advances in Neural Information Processing Systems*, pages 5756–5766, 2017.

[45] Mingze Xu, Mingfei Gao, Yi-Ting Chen, Larry S Davis, and David J Crandall. Temporal recurrent networks for online action detection. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5532–5541, 2019.

[46] Tianfan Xue, Jiajun Wu, Katherine Bouman, and Bill Freeman. Visual dynamics: Probabilistic future frame synthesis via cross convolutional networks. In *Advances in neural information processing systems*, pages 91–99, 2016.