A Variational Auto-Encoder Model for Stochastic Point Processes

Nazanin Mehrasa[1,3], Akash Abdu Jyothi[1,3], Thibaut Durand[1,3], Jiawei He[1,3], Leonid Sigal[2,3], Greg Mori[1,3]
[1]Simon Fraser University [2]University of British Columbia [3]Borealis AI
{nmehrasa, aabdujyo, tdurand, jha203}@sfu.ca lsigal@cs.ubc.ca mori@cs.sfu.ca

Abstract

We propose a novel probabilistic generative model for action sequences. The model is termed the Action Point Process VAE (APP-VAE), a variational auto-encoder that can capture the distribution over the times and categories of action sequences. Modeling the variety of possible action sequences is a challenge, which we show can be addressed via the APP-VAE's use of latent representations and non-linear functions to parameterize distributions over which event is likely to occur next in a sequence and at what time. We empirically validate the efficacy of APP-VAE for modeling action sequences on the MultiTHUMOS and Breakfast datasets.

1. Introduction

Anticipatory reasoning to model the evolution of action sequences over time is a fundamental challenge in human activity understanding. The crux of the problem in making predictions about the future is the fact that for interesting domains, the future is uncertain – given a history of actions such as those depicted in Fig. 1, the distribution over future actions has substantial entropy.

In this work, we propose a powerful generative approach that can effectively model the categorical and temporal variability comprising action sequences. Much of the work in this domain has focused on taking frame level data of video as input in order to predict the actions or activities that may occur in the immediate future. There has also been recent interest on the task of predicting the sequence of actions that occur farther into the future [6, 32, 1].

Time series data often involves regularly spaced data points with interesting events occurring sparsely across time. This is true in case of videos where we have a regular frame rate but events of interest are present only in some frames that are infrequent. We hypothesize that in order to model future events in such a scenario, it is beneficial to consider the history of sparse events (action categories and their temporal occurrence in the above example) alone, instead of regularly spaced frame data. While the history of frames contains rich information over and above the sparse event history, we can possibly create a model for future events occurring farther into the future by choosing to only model the sparse sequence of events. This approach also allows us to model high-level semantic meaning in the time series data that can be difficult to discern from low-level data points that are regular across time.

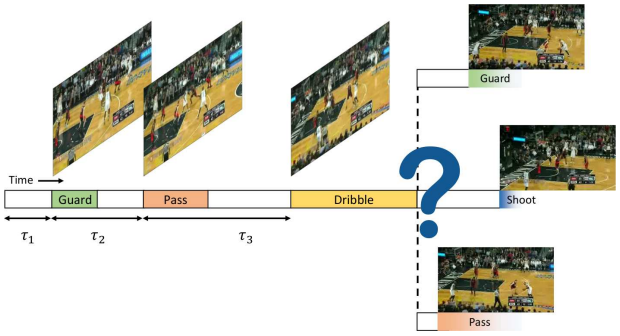

Figure 1. It is difficult to make predictions, especially about the future. Given a history of past actions, multiple actions are possible in the future. We focus on the problem of learning a distribution over the future actions – *what* are the possible action categories and *when* will they start.

Our model is formulated in the variational auto-encoder (VAE) [15] paradigm, a powerful class of probabilistic models that facilitate generation and the ability to model complex distributions. We present a novel form of VAE for action sequences under a point process approach. This approach has a number of advantages, including a probabilistic treatment of action sequences to allow for likelihood evaluation, generation, and anomaly detection.

Contribution. The contributions of this work center around the APP-VAE (Action Point Process VAE), a novel generative model for asynchronous time action sequences. The contributions of this paper include:

- A novel formulation for modeling point process data within the variational auto-encoder paradigm.
- Conditional prior models for encoding asynchronous time data.

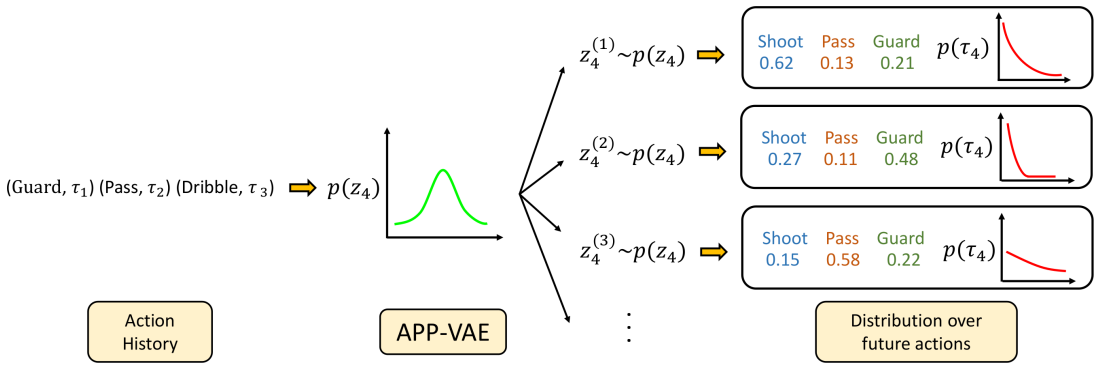

Figure 2. Given the history of actions, APP-VAE generates a distribution over possible actions in the next step. APP-VAE can recurrently perform this operation to model diverse sequences of actions that may follow. The figure shows the distributions for the fourth action in a basketball game given the history of first three actions.

- A probabilistic model for jointly capturing uncertainty in which actions will occur and when they will happen.

2. Related Work

Activity Prediction. Most activity prediction tasks are frame-based, *i.e*. the input to the model is a sequence of frames before the action starts and the task is predict what will happen next. Lan *et al*. [18] predict future actions from hierarchical representations of short clips by having different classifiers at each level in a max-margin framework. Mahmud *et al*. [20] jointly predicts future activity as well as its starting time by a multi-streams framework. Each streams tries to catch different features for having a richer feature representation for future prediction: One stream for visual information, one for previous activities and the last one focusing on the last activity.

Farha *et al*. [1] proposed a framework for predicting the action categories of a sequence of future activities as well as their starting and ending time. They proposed two deterministic models, one using a combination of RNN and HMM and the other one is a CNN predicting a matrix which future actions are encoded in it.

Asynchronous Action Prediction. We focus on the task of predicting future action given a sequence of previous actions that are asynchronous in time. Du *et al*. [6] proposed a recurrent temporal model for learning the next activity timing and category given the history of previous actions. Their recurrent model learns a non-linear map of history to the intensity function of a temporal point process framework. Zhong *et al*. [32] also introduced a hierarchical recurrent network model for future action prediction for modeling future action timing and category. Their model takes frame-level information as well as sparse high-level events information in the history to learn the intensity function of a temporal point process. Xiao *et al*. [28] introduced an intensity-free generative method for temporal point process. The generative part of their model is an extension of Wasserstein GAN in the context of temporal point process for learning to generate sequences of action.

Early Stage Action Prediction. Our work is related to early stage action prediction. This task refers to predicting the action given the initial frames of the activity [19, 10, 25]. Our task is different from early action prediction, because the model doesn't have any information about the action while predicting it. Recently Yu *et al*. [31] used variational auto-encoder to learn from the frames in the history and transfer them into the future. Sadegh Aliakbarian *et al*. [24] combine context and action information using a multi-stage LSTM model to predict future action. The model is trained with a loss function which encourages the model to predict action with few observations. Gao *et al*. [7] proposed to use a Reinforced Encoder-Decoder network for future activity prediction. Damen *et al*. [3] proposed a semi-supervised variational recurrent neural network to model human activity including classification, prediction, detection and anticipation of human activities.

Video Prediction. Video prediction has recently been studied in several works. Denton and Fergus [5] use a variational auto-encoder framework with a learned prior to generate future video frames. He *et al*. [9] also proposed a generative model for future prediction. They structure the latent space by adding control features which makes the model able to control generation. Vondrick *et al*. [27] uses adversarial learning for generating videos of future with transforming the past pixels. Patraucean et al. [23] describe a spatio-temporal auto-encoder that predicts optical flow as a dense map, using reconstruction in its learning criterion. Villegas et al. [26] propose a hierarchical approach to pixel-level video generation, reasoning over body pose before rendering into a predicted future frame.

3. Asynchronous Action Sequence Modeling

We first introduce some notations and the problem definition. Then we review the VAE model and temporal

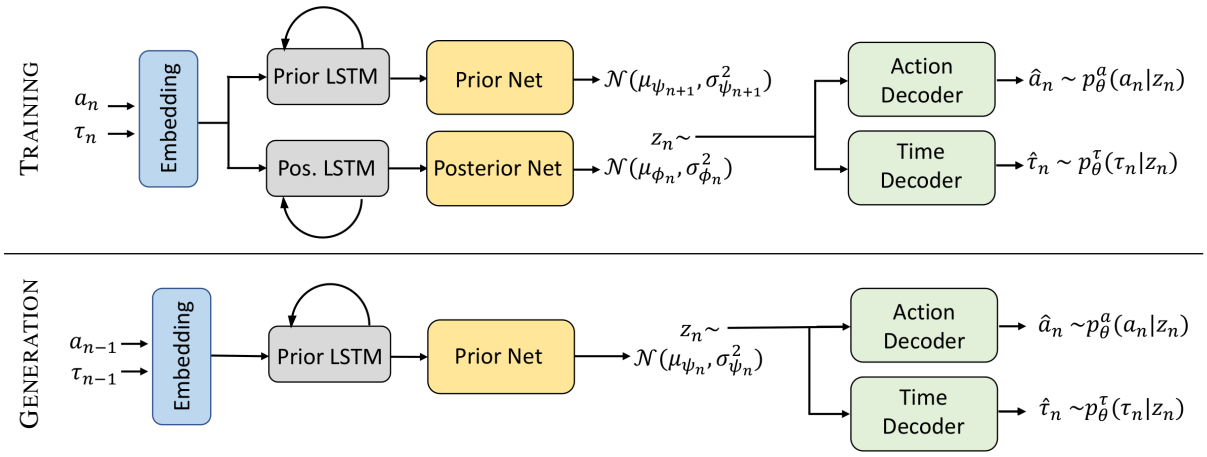

Figure 3. Our proposed recurrent VAE model for asynchronous action sequence modeling. At each time step, the model uses the history of actions and inter-arrival times to generate a distribution over latent codes, a sample of which is then decoded into two probability distributions for the next action: one over possible action labels and one over the inter arrival time.

point process that are used in our model. Subsequently, we present our model in detail and how it is trained.

Problem definition. The input is a sequence of actions $x_{1:n} = (x_1, \ldots, x_n)$ where x_n is the n-th action. The action $x_n = (a_n, \tau_n)$ is represented by the action category $a_n \in \{1, 2, \ldots, K\}$ (K discrete action classes) and the inter-arrival time $\tau_n \in \mathbb{R}^+$. The inter-arrival time is the difference between the starting time of action x_{n-1} and x_n. We formulate the asynchronous action distribution modeling task as follows: given a sequence of actions $x_{1:n-1}$, the goal is to produce a distribution over what action a_n will happen next, and the inter arrival time τ_n. We aim to develop probabilistic models to capture the uncertainty over these what and when questions of action sequence modeling.

3.1. Background: Base Models

Variational Auto-Encoders (VAEs). A VAE [15] describes a generative process with simple prior $p_\theta(z)$ (usually chosen to be a multivariate Gaussian) and complex likelihood $p_\theta(x|z)$ (the parameters of which are produced by neural networks). x and z are observed and latent variables, respectively. Approximating the intractable posterior $p_\theta(z|x)$ with a recognition neural network $q_\phi(z|x)$, the parameters of the generative model θ as well as the recognition model ϕ can be jointly optimized by maximizing the evidence lower bound $\mathcal{L}$ on the marginal likelihood $p_\theta(x)$:

$$\begin{aligned} \log p_\theta(x) &= \mathrm{KL}(q_\phi \| p_\theta) + \mathcal{L}(\theta, \phi) \\ &\geq \mathcal{L}(\theta, \phi) = -\mathbb{E}_{q_\phi}\left[\log \frac{q_\phi(z|x)}{p_\theta(z, x)}\right]. \end{aligned} \tag{1}$$

Recent works expand VAEs to time-series data including video [2, 5, 9], text [4, 12], or audio [30]. A popular design choice of such models is the integration of a per time-step VAE with RNN/LSTM temporal modelling. The ELBO thus becomes a summation of time-step-wise variational lower bound[1]:

$$\begin{aligned} \mathcal{L}(\theta, \phi, \psi) &= \sum_{n=1}^{N} \Big[\mathbb{E}_{q_\phi(z_{1:n}|x_{1:n})}\left[\log p_\theta(x_n|x_{1:n-1}, z_{1:n})\right] \\ &- \mathrm{KL}(q_\phi(z_n|x_{1:n}) || p_\psi(z_n|x_{1:n-1}))\Big]. \end{aligned} \tag{2}$$

with a "prior" $p_\psi(z_n|x_{1:n-1})$ that evolves over the N time steps used.

Temporal point process. A temporal point process is a stochastic model used to capture the inter-arrival times of a series of events. A temporal point process is characterized by the conditional intensity function $\lambda(\tau_n|x_{1:n-1})$, which is conditioned on the past events $x_{1:n-1}$ (*e.g.* action in this work). The conditional intensity encodes instantaneous probabilities at time τ. Given the history of $n-1$ past actions, the probability density function for the time of the next action is:

$$f(\tau_n|x_{1:n-1}) = \lambda(\tau_n|x_{1:n-1}) e^{-\int_0^{\tau_n} \lambda(u|x_{1:n-1})\ \mathrm{d}u} \tag{3}$$

The Poisson process [16] is a popular temporal point process, which assumes that events occur independent of one another. The conditional intensity is $\lambda(\tau_n|x_{1:n-1}) = \lambda$ where λ is a positive constant. More complex conditional intensities have been proposed like Hawkes Process [8] and Self-Correcting Process [13]. All these conditional intensity function seek to capture some forms of dependency on

[1]Note that variants exist, depending on the exact form of the recurrent structure and its VAE instantiation.

the past action. However, in practice the true model of the dependencies is never known [21] and the performance depend on the design of the conditional intensity. In this work, we learn a recurrent model that estimates the conditional intensity based on the history of actions.

3.2. Proposed Approach

We propose a generative model for asynchronous action sequence modeling using the VAE framework. Figure 3 shows the architecture of our model. Overall, the input sequence of actions and inter arrival times are encoded using a recurrent VAE model. At each step, the model uses the history of actions to produce a distribution over latent codes z_n, a sample of which is then decoded into two probability distributions: one over the possible action categories and another over the inter-arrival time for the next action. We now detail our model.

Model. At time step n during training, the model takes as input the action x_n, which is the target of the prediction model, and the history of past actions $x_{1:n-1}$. These inputs are used to compute a conditional distribution $q_\phi(z_n|x_{1:n})$ from which a latent code z_n is sampled. Since the true distribution over latent variables z_n is intractable we rely on a time-dependent inference network $q_\phi(z_n|x_{1:n})$ that approximates it with a conditional Gaussian distribution $\mathcal{N}(\mu_{\phi_n}, \sigma^2_{\phi_n})$. To prevent z_n from just copying x_n, we force $q_\phi(z_n|x_{1:n})$ to be close to the prior distribution $p(z_n)$ using a KL-divergence term. Usually in VAE models, $p(z_n)$ is a fixed Gaussian $\mathcal{N}(0, I)$. But a drawback of using a fixed prior is that samples at each time step are drawn randomly, and thus ignore temporal dependencies present between actions. To overcome this problem, a solution is to learn a prior that varies across time, being a function of all past actions except the current action $p_\psi(z_{n+1}|x_{1:n})$. Both prior and approximate posterior are modelled as multivariate Gaussian distributions with diagonal covariance with parameters as shown below:

$$q_\phi(z_n|x_{1:n}) = \mathcal{N}(\mu_{\phi_n}, \sigma^2_{\phi_n}) \tag{4}$$

$$p_\psi(z_{n+1}|x_{1:n}) = \mathcal{N}(\mu_{\psi_{n+1}}, \sigma^2_{\psi_{n+1}}) \tag{5}$$

At step n, both posterior and prior networks observe actions $x_{1:n}$ but the posterior network outputs the parameters of a conditional Gaussian distribution for the current action x_n whereas the prior network outputs the parameters of a conditional Gaussian distribution for the next action x_{n+1}.

At each time-step during training, a latent variable z_n is drawn from the posterior distribution $q_\phi(z_n|x_{1:n})$. The output action $\hat{x}_n$ is then sampled from the distribution $p_\theta(x_n|z_n)$ of our conditional generative model which is parameterized by θ. For mathematical convenience, we assume the action category and inter-arrival time are conditionally independent given the latent code z_n:

$$p_\theta(x_n|z_n) = p_\theta(a_n, \tau_n|z_n) = p^a_\theta(a_n|z_n)p^\tau_\theta(\tau_n|z_n) \tag{6}$$

where $p^a_\theta(a_n|z_n)$ (resp. $p^\tau_\theta(\tau_n|z_n)$) is the conditional generative model for action category (resp. inter-arrival time). This is a standard assumption in event prediction [6, 32]. The sequence model generates two probability distributions: (i) a categorical distribution over the action categories and (ii) a temporal point process distribution over the inter-arrival times for the next action.

The distribution over action categories is modeled with a multinomial distribution when a_n can only take a finite number of values:

$$p^a_\theta(a_n = k|z_n) = p_k(z_n) \quad \text{and} \quad \sum_{k=1}^{K} p_k(z_n) = 1 \tag{7}$$

where $p_k(z_n)$ is the probability of occurrence of action k, and K is the total number of action categories.

The inter-arrival time is assumed to follow an exponential distribution parameterized by $\lambda(z_n)$, similar to a standard temporal point process model:

$$p^\tau_\theta(\tau_n|z_n) = \begin{cases} \lambda(z_n)e^{-\lambda(z_n)\tau_n} & \text{if } \tau_n \geq 0 \\ 0 & \text{if } \tau_n < 0 \end{cases} \tag{8}$$

where $p^\tau_\theta(\tau_n|z_n)$ is a probability density function over random variable τ_n and $\lambda(z_n)$ is the intensity of the process, which depends on the latent variable sample z_n.

Learning. We train the model by optimizing the variational lower bound over the entire sequence comprised of N steps:

$$\mathcal{L}_{\theta,\phi}(x_{1:N}) = \sum_{n=1}^{N} (\mathbb{E}_{q_\phi(z_n|x_{1:n})}[\log p_\theta(x_n|z_n)] - D_{KL}(q_\phi(z_n|x_{1:n})||p_\psi(z_n|x_{1:n-1}))) \tag{9}$$

Because the action category and inter-arrival time are conditionally independent given the latent code z_n, the log-likelihood term can be written as follows:

$$\mathbb{E}_{q_\phi(z_n|x_{1:n})}[\log p_\theta(x_n|z_n)] = \mathbb{E}_{q_\phi(z_n|x_{1:n})}[\log p^a_\theta(a_n|z_n)] + \mathbb{E}_{q_\phi(z_n|x_{1:n})}[\log p^\tau_\theta(\tau_n|z_n)] \tag{10}$$

Given the form of p^a_θ the log-likelihood term reduces to a cross entropy between the predicted action category distribution $p^a_\theta(a_n|z_n)$ and the ground truth label a^*_n. Given the ground truth inter-arrival time τ^*_n, we compute its log-likelihood over a small time interval Δ_τ under the predicted distribution.

$$\log \left[\int_{\tau^*_n}^{\tau^*_n + \Delta_\tau} p^\tau_\theta(\tau_n|z_n)\, \mathrm{d}\tau_n \right] = \log(1 - e^{-\lambda(z_n)\Delta_\tau}) - \lambda(z_n)\tau^*_n \tag{11}$$

We use the re-parameterization trick [15] to sample from the encoder network q_ϕ.

Generation. The goal is to generate the next action $\hat{x}_n = (\hat{a}_n, \hat{\tau}_n)$ given a sequence of past actions $x_{1:n-1}$. The generation process is shown on the bottom of Figure 3. At test time, an action at step n is generated by first sampling z_n from the prior. The parameters of the prior distribution are computed based on the past $n-1$ actions $x_{1:n-1}$. Then, an action category $\hat{a}_n$ and inter-arrival time $\hat{\tau}_n$ are generated as follows:

$$\hat{a}_n \sim p^a_\theta(a_n|z_n) \qquad \hat{\tau}_n \sim p^\tau_\theta(\tau_n|z_n) \tag{12}$$

Architecture. We now describe the architecture of our model in detail. At step n, the current action x_n is embedded into a vector representation x^{emb}_n with a two-step embedding strategy. First, we compute a representation for the action category (a_n) and the inter-arrival time (τ_n) separately. Then, we concatenate these two representations and compute a new representation x^{emb}_n of the action.

$$a^{emb}_n = f^{emb}_a(a_n) \qquad \tau^{emb}_n = f^{emb}_\tau(\tau_n) \tag{13}$$

$$x^{emb}_n = f^{emb}_{a,\tau}([a^{emb}_n, \tau^{emb}_n]) \tag{14}$$

We use a 1-hot encoding to represent the action category label a_n. Then, we have two branches: one to estimate the parameters of the posterior distribution and another to estimate the parameters of the prior distribution. The network architecture of these two branches is similar but we use separate networks because the prior and the posterior distribution capture different information. Each branch has a Long Short Term Memory (LSTM) [11] to encode the current action and the past actions into a vector representation:

$$h^{post}_n = LSTM_\phi(x^{emb}_n, h^{post}_{n-1}) \tag{15}$$

$$h^{prior}_n = LSTM_\psi(x^{emb}_n, h^{prior}_{n-1}) \tag{16}$$

Recurrent networks turn variable length sequences into meaningful, fixed-sized representations. The output of the posterior LSTM h^{post}_n (resp. prior LSTM h^{prior}_n) is passed into a posterior (also called inference) network f^{post}_ϕ (resp. prior network f^{prior}_ψ) that outputs the parameters of the Gaussian distribution:

$$\mu_{\phi_n}, \sigma^2_{\phi_n} = f^{post}_\phi(h^{post}_n) \tag{17}$$

$$\mu_{\psi_n}, \sigma^2_{\psi_n} = f^{prior}_\psi(h^{prior}_n) \tag{18}$$

Then, a latent variable z_n is sampled from the posterior (or prior during testing) distribution and is fed to the decoder networks for generating distributions over the action category a_n and inter-arrival time τ_n.

The decoder network for action category $f^a_\theta(z_n)$ is a multi-layer perceptron with a softmax output to generate the probability distribution in Eq. 7:

$$p^a_\theta(a_n|z_n) = f^a_\theta(z_n) \tag{19}$$

The decoder network for inter-arrival time $f^\tau_\theta(z_n)$ is another multi-layer perceptron, producing the parameter for the point process model for temporal distribution in Eq. 8:

$$\lambda(z_n) = f^\tau_\theta(z_n) \tag{20}$$

During training, the parameters of all the networks are jointly learned in an end-to-end fashion.

4. Experiments

Datasets. We performed experiments using APP-VAE on two action recognition datasets. We use the standard training and testing sets for each.

MultiTHUMOS Dataset [29] is a challenging dataset for action recognition, containing 400 videos of 65 different actions. On average, there are 10.5 action class labels per video and 1.5 actions per frame.

Breakfast Dataset [17] contains 1712 videos of breakfast preparation for 48 action classes. The actions are performed by 52 people in 18 different kitchens.

Architecture details. The APP-VAE model architecture is shown in Fig. 3. Action category and inter-arrival time inputs are each passed through 2 layer MLPs with ReLU activation. They are then concatenated and followed with a linear layer. Hidden state of prior and posterior LSTMs is 128. Both prior and posterior networks are 2 layer MLPs, with ReLU activation after the first layer. Dimension of the latent code is 256. Action decoder is a 3 layer MLP with ReLU at the first two layers and softmax for the last one. The time decoder is also a 3 layer MLP with ReLU at the first two layers, with an exponential non-linearity applied to the output to ensure the parameter of the point process is positive.

Implementation details. The models are implemented with PyTorch [22] and are trained using the Adam [14] optimizer for 1,500 epochs with batch size 32 and learning rate 0.001. We split the standard training set of both datasets into training and validation sets containing 70% and 30% of samples respectively. We select the best model during training based on the model loss (Eq. 10) on the validation set.

Baselines. We compare APP-VAE with the following models for action prediction tasks.

Dataset	Model	Stoch. Var.	LL
Breakfast	APP-LSTM	-	-6.668
	APP-VAE w/o Learned Prior	✓	≥-9.427
	APP-VAE	✓	≥**-5.944**
MultiTUHMOS	APP-LSTM	-	-4.190
	APP-VAE w/o Learned Prior	✓	≥-5.344
	APP-VAE	✓	≥**-3.838**

Table 1. Comparison of log-likelihood on Breakfast and MultiTHUMOS datasets.

- *Time Deterministic LSTM (TD-LSTM).* This is a vanilla LSTM model that is trained to predict the next action category and the inter-arrival time, comparable with the model proposed by Farha *et al.* [1]. This model directly predicts the inter-arrival time and not the distribution over it. TD-LSTM uses the same encoder network as APP-VAE. We use cross-entropy loss for action category output and perform regression over inter-arrival time using mean squared error (MSE) loss similar to [1].

- *Action Point Process LSTM (APP-LSTM).* This baseline predicts the inter-arrival time distribution similar to APP-VAE. The model uses the same reconstruction loss function as in the VAE model – cross entropy loss for action category and negative log-likelihood (NLL) loss for inter-arrival time. APP-LSTM does not have the stochastic latent code that allows APP-VAE to model diverse distributions over action category and inter-arrival time. Our APP-LSTM baseline encompasses Du *et al.* [6]'s work. The only difference is the way we model the intensity function (IF). Du *et al.* [6] defines IS explicitly as a function of time. This design choice has been investigated in Zhong *et al.* [32]; an implicit intensity function is shown to be superior and thus adapted in our APP-LSTM baseline.

Metrics. We use log-likelihood (LL) to compare our model with the APP-LSTM. We also report accuracy of action category prediction and mean absolute error (MAE) of inter-arrival time prediction. We calculate accuracy by comparing the most probable action category from the model output with the ground truth category. To calculate MAE, we use the expected inter-arrival time under the predicted distribution $p_\theta^\tau(\tau_n|z_n)$:

$$\mathbb{E}_{p_\theta^\tau(\tau_n|z_n)}[\tau_n] = \int_0^\infty \tau_n \cdot p_\theta^\tau(\tau_n|z_n)\mathrm{d}\tau_n = \frac{1}{\lambda(z_n)} \qquad (21)$$

The expected value $\frac{1}{\lambda(z_n)}$ and the ground truth inter-arrival time are used to compute MAE.

4.1. Experiment Results

We discuss quantitative and qualitative results from our experiments. All quantitative experiments are performed by teacher forcing methodology *i.e.* for each step in the sequence of actions, the models are fed the ground truth history of actions, and likelihood and/or other metrics for the next action are measured.

Quantitative results. Table 1 shows experimental results that compare APP-VAE with the APP-LSTM. To estimate the log-likelihood (LL) of our model, we draw 1500 samples from the approximate posterior distribution, following the standard approach of importance sampling. APP-VAE outperforms the APP-LSTM on both MultiTHUMOS and Breakfast datasets. We believe that this is because the APP-VAE model is better in modeling the complex distribution over future actions.

Table 2 shows accuracy and MAE in predicting the future action given the history of previous actions. APP-VAE outperforms TD-LSTM and APP-LSTM under both the metrics. For each step in the sequence we draw 1500 samples from the prior distribution that models the next step action. Given the output distributions, we select the action category with the maximum probability as the predicted action, and the expected value of inter-arrival time as the predicted inter-arrival time. Out of 1500 predictions, we select the most frequent action as the model prediction for that time step, and compute inter-arrival time by averaging over the corresponding time values.

Table 1 and 2 also show the comparison of our model with the case where the prior is fixed in all of the time-steps. In this experiment, we fixed the prior to the standard normal distribution $\mathcal{N}(0, I)$. We can see that the learned prior variant outperforms the fixed prior variant consistently across all datasets. The model with the fixed prior does not perform well because it learns to predict the majority action class and average inter-arrival time of the training set, ignoring the history of any input test sequence.

In addition to the above strategy of selecting the mode action at each step, we also report action category accuracy and MAE obtained by averaging over predictions of all 1500 samples. We summarize these results in Table 4.

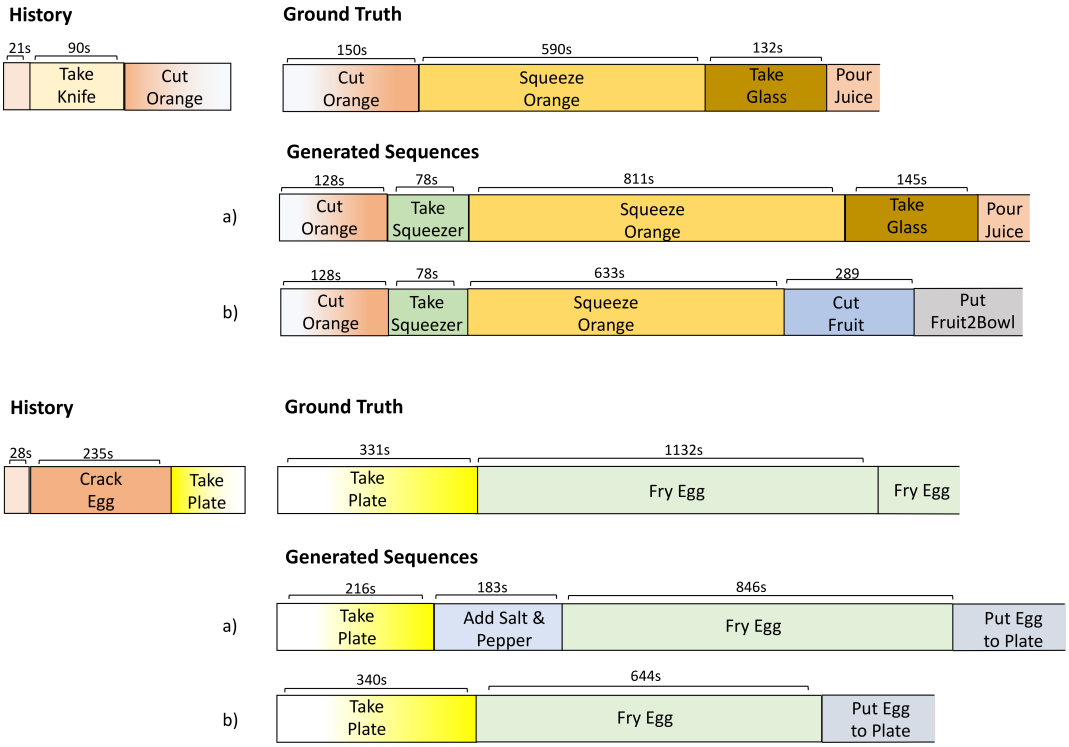

Figure 4. Examples of generated sequences. Given the history (shown at left), we generate a distribution over latent code z_n for the subsequent time step. A sample is drawn from this distribution, and decoded into distributions over action category and time, from which a next action/time pair by selecting the action with the highest probability and computing the expectation of the generated distribution over τ (Equation 21). This process is repeated to generate a sequence of actions. Two such sampled sequences (a) and (b) are shown for each history, and compared to the respective ground truth sequence (in line with history row). We can see that APP-VAE is capable of generating diverse and plausible action sequences.

Dataset	Model	Time Loss	stoch. var.	↑ accuracy	↓ MAE
Breakfast	TD-LSTM	MSE	-	53.64	173.76
	APP-LSTM	NLL	-	61.39	152.17
	APP-VAE w/o Learned Prior	NLL	✓	27.09	270.75
	APP-VAE	NLL	✓	**62.20**	**142.65**
MultiTUHMOS	TD-LSTM	MSE	-	29.74	2.33
	APP-LSTM	NLL	-	36.31	1.99
	APP-VAE w/o Learned Prior	NLL	✓	8.79	2.02
	APP-VAE	NLL	✓	**39.30**	**1.89**

Table 2. Accuracy of action category prediction and Mean Absolute Error (MAE) of inter-arrival time prediction of all model variants. Arrows show whether lower (↓) or higher (↑) scores are better.

We next explore the architecture of our model by varying the sizes of the latent variable. Table 5 shows the log-likelihood of our model for different sizes of the latent variable. We see that as we increase the size of the latent variable, we can model a more complex latent distribution which results in better performance.

Qualitative Results. Fig. 4 shows examples of diverse future action sequences that are generated by APP-VAE given the history. For different provided histories, sampled sequences of actions are shown. We note that the overall duration and sequence of actions on the Breakfast Dataset are reasonable. Variations, e.g. taking the juice squeezer before using it, adding salt and pepper before cooking eggs, are plausible alternatives generated by our model.

Fig. 5 visualizes a traversal on one of the latent codes for three different sequences by uniformly sampling one z dimension over $[\mu - 5\sigma, \mu + 5\sigma]$ while fixing others to their sampled values. As shown, this dimension correlates closely with the action *add_saltnpepper*, *strifry_egg*

	Test sequences with high likelihood
1	NoHuman, CliffDiving, Diving, Jump, BodyRoll, CliffDiving, Diving, Jump, BodyRoll, CliffDiving, Diving, Jump, BodyRoll, BodyContract, Run, CliffDiving, Diving, Jump, ..., BodyRoll, CliffDiving, Diving, BodyContract, CliffDiving, Diving, CliffDiving, Diving, CliffDiving, Diving, Jump, CliffDiving, Diving, Walk, Run, Jump, Jump, Run, Jump
2	CleanAndJerk, PickUp, BodyContract, Squat, StandUp, BodyContract, Squat, CleanAndJerk, PickUp, StandUp, BodyContract, Squat, CleanAndJerk, PickUp, StandUp, Drop, BodyContract, Squat, PickUp, ..., Squat, StandUp, Drop, BodyContract, Squat, BodyContract, Squat, BodyContract, Squat, BodyContract, Squat, BodyContract, Squat, NoHuman
	Test sequences with low likelihood
1	NoHuman, TalkToCamera, GolfSwing, GolfSwing, GolfSwing, GolfSwing, NoHuman
2	NoHuman, HammerThrow, TalkToCamera, CloseUpTalkToCamera, HammerThrow, HammerThrow, HammerThrow, TalkToCamera, ..., HammerThrow, HammerThrow, HammerThrow, HammerThrow, HammerThrow, HammerThrow, HammerThrow, HammerThrow, HammerThrow, HammerThrow, HammerThrow, HammerThrow

Table 3. Example of test sequences with high and low likelihood according to our learned model

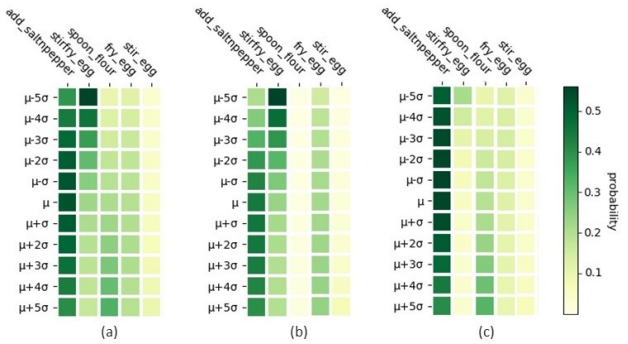

Figure 5. **Latent Code Manipulation.** The history + ground-truth label of future action for the sub-figures are: (a) "SIL, crack_egg"→"add_saltnpepper", (b) "SIL, take_plate, crack_egg"→ "add_saltnpepper" and (c) "SIL, pour_oil, crack_egg"→"add_saltnpepper".

Dataset	Model	Acc	MAE
Breakfast	APP-VAE - avg	59.02	145.95
	APP-VAE - mode	62.20	142.65
MultiTUHMOS	APP-VAE - avg	35.23	1.96
	APP-VAE - mode	39.30	1.89

Table 4. Accuracy (Acc) and Mean Absolute Error (MAE) under mode and averaging over samples.

and *fry_egg*.

We further qualitatively examine the ability of the model to score the likelihood of individual test samples. We sort the test action sequences according to the average per time-step likelihood estimated by drawing 1500 samples from the approximate posterior distribution following the importance sampling approach. High scoring sequences should be those that our model deems as "normal" while low scoring sequences those that are unusual. Tab. 3 shows some example of sequences with low and high likelihood on the MultiTHUMOS dataset. We note that a regular, structured sequence of actions such as jump, body roll, cliff diving for a diving action or body contract, squat, clean and jerk for a weightlifting action receives high likelihood. However, repeated hammer throws or golf swings with no set up actions receives a low likelihood.

Latent size	32	64	128	256	512
LL (≥)	-4.486	-3.947	-3.940	-3.838	-4.098

Table 5. Log-likelihood for APP-VAE with different latent variable dimensionality on MultiTHUMOS.

Finally we compare asynchronous APP-LSTM with a synchronous variant (with constant frame rate) on Breakfast dataset. The synchronous model predicts actions one step at a time and the sequence is post-processed to infer the duration of each action. The performance is significantly worse for both MAE time (152.17 vs 1459.99) and action prediction accuracy (61.39% vs 28.24%). A plausible explanation is that LSTMs cannot deal with very long-term dependencies.

5. Conclusion

We presented a novel probabilistic model for point process data – a variational auto-encoder that captures uncertainty in action times and category labels. As a generative model, it can produce action sequences by sampling from a prior distribution, the parameters of which are updated based on neural networks that control the distributions over the next action type and its temporal occurrence. The model can also be used to analyze given input sequences of actions to determine the likelihood of observing particular sequences. We demonstrate empirically that the model is effective for capturing the uncertainty inherent in tasks such as action prediction and anomaly detection.

References

[1] 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 (CVPR)*, 2018. 1, 2, 6

[2] Mohammad Babaeizadeh, Chelsea Finn, Dumitru Erhan, Roy H. Campbell, and Sergey Levine. Stochastic Variational Video Prediction. In *International Conference on Learning Representations (ICLR)*, 2018. 3

[3] Judith Bütepage, Hedvig Kjellström, and Danica Kragic. Classify, predict, detect, anticipate and synthesize: Hierarchical recurrent latent variable models for human activity modeling. *arXiv preprint arXiv:1809.08875*, 2018. 2

[4] Junyoung Chung, Kyle Kastner, Laurent Dinh, Kratarth Goel, Aaron C Courville, and Yoshua Bengio. A recurrent latent variable model for sequential data. In *Advances in neural information processing systems*, pages 2980–2988, 2015. 3

[5] Emily Denton and Rob Fergus. Stochastic Video Generation with a Learned Prior. In *International Conference on Machine Learning (ICML)*, 2018. 2, 3

[6] Nan Du, Hanjun Dai, Rakshit Trivedi, Utkarsh Upadhyay, Manuel Gomez-Rodriguez, and Le Song. Recurrent marked temporal point processes: Embedding event history to vector. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016. 1, 2, 4, 6

[7] Jiyang Gao, Zhenheng Yang, and Ram Nevatia. RED: reinforced encoder-decoder networks for action anticipation. *CoRR*, abs/1707.04818, 2017. 2

[8] Alan G Hawkes. Spectra of some self-exciting and mutually exciting point processes. *Biometrika*, 1971. 3

[9] Jiawei He, Andreas Lehrmann, Joseph Marino, Greg Mori, and Leonid Sigal. Probabilistic video generation using holistic attribute control. In *The European Conference on Computer Vision (ECCV)*, September 2018. 2, 3

[10] M. Hoai and F. De la Torre. Max-margin early event detectors. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012. 2

[11] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Comput.*, 1997. 5

[12] Zhiting Hu, Zichao Yang, Xiaodan Liang, Ruslan Salakhutdinov, and Eric P Xing. Toward controlled generation of text. In *International Conference on Machine Learning*, pages 1587–1596, 2017. 3

[13] Valerie Isham and Mark Westcott. A self-correcting point process. *Stochastic Processes and their Applications*, 1979. 3

[14] Diederick P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, 2015. 5

[15] Diederik P Kingma and Max Welling. Auto-Encoding Variational Bayes. In *International Conference on Learning Representations (ICLR)*, 2014. 1, 3, 5

[16] J. F. C. Kingman. *Poisson processes*. 1993. 3

[17] Hilde Kuehne, Ali Arslan, and Thomas Serre. The Language of Actions: Recovering the Syntax and Semantics of Goal-Directed Human Activities. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014. 5

[18] Tian Lan, Tsung-Chuan Chen, and Silvio Savarese. A hierarchical representation for future action prediction. In *European Conference on Computer Vision (ECCV)*, 2014. 2

[19] S. Ma, L. Sigal, and S. Sclaroff. Learning activity progression in lstms for activity detection and early detection. In *cvpr*, 2016. 2

[20] 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 (ICCV)*, 2017. 2

[21] Hongyuan Mei and Jason Eisner. The Neural Hawkes Process: A Neurally Self-Modulating Multivariate Point Process. In *Advances in Neural Information Processing Systems (NIPS)*, 2017. 4

[22] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. In *Advances in Neural Information Processing Systems (NIPS)*, 2017. 5

[23] Viorica Pătrăucean, Ankur Handa, and Roberto Cipolla. Spatio-temporal video autoencoder with differentiable memory. In *International Conference on Learning Representations (ICLR) Workshop*, 2016. 2

[24] Mohammad Sadegh Aliakbarian, Fatemeh Sadat Saleh, Mathieu Salzmann, Basura Fernando, Lars Petersson, and Lars Andersson. Encouraging LSTMs to Anticipate Actions Very Early. In *IEEE International Conference on Computer Vision (ICCV)*, 2017. 2

[25] Yuge Shi, Basura Fernando, and Richard Hartley. Action anticipation with rbf kernelized feature mapping rnn. In *The European Conference on Computer Vision (ECCV)*, September 2018. 2

[26] Ruben Villegas, Jimei Yang, Yuliang Zou, Sungryull Sohn, Xunyu Lin, and Honglak Lee. Learning to Generate Long-term Future via Hierarchical Prediction. In *International Conference on Machine Learning (ICML)*, 2017. 2

[27] Carl Vondrick and Antonio Torralba. Generating the future with adversarial transformers. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017. 2

[28] Shuai Xiao, Mehrdad Farajtabar, Xiaojing Ye, Junchi Yan, Le Song, and Hongyuan Zha. Wasserstein learning of deep generative point process models. In *Advances in Neural Information Processing Systems (NIPS)*, 2017. 2

[29] Serena Yeung, Olga Russakovsky, Ning Jin, Mykhaylo Andriluka, Greg Mori, and Li Fei-Fei. Every Moment Counts: Dense Detailed Labeling of Actions in Complex Videos. *International Journal of Computer Vision (IJCV)*, 2017. 5

[30] Li Yingzhen and Stephan Mandt. Disentangled sequential autoencoder. In *International Conference on Machine Learning*, 2018. 3

[31] Runsheng Yu, Zhenyu Shi, and Laiyun Qing. Unsupervised learning aids prediction: Using future representation learning variantial autoencoder for human action prediction. *CoRR*, abs/1711.09265, 2017. 2

[32] Y. Zhong, B. Xu, G.-T. Zhou, L. Bornn, and G. Mori. Time Perception Machine: Temporal Point Processes for the When, Where and What of Activity Prediction. *ArXiv e-prints*, Aug. 2018. 1, 2, 4, 6