On Diverse Asynchronous Activity Anticipation

He Zhao and Richard P. Wildes

York University, Toronto, Ontario, Canada
{zhufl, wildes}@cse.yorku.ca

Abstract. We investigate the joint anticipation of long-term activity labels and their corresponding times with the aim of improving both the naturalness and diversity of predictions. We address these matters using Conditional Adversarial Generative Networks for Discrete Sequences. Central to our approach is a reexamination of the unavoidable sample quality vs. diversity tradeoff of the recently emerged Gumbel-Softmax relaxation based GAN on discrete data. In particular, we ameliorate this trade-off with a simple but effective sample distance regularizer. Moreover, we provide a unified approach to inference of activity labels and their times so that a single integrated optimization succeeds for both. With this novel approach in hand, we demonstrate the effectiveness of the resulting discrete sequential GAN on multimodal activity anticipation. We evaluate the approach on three standard datasets and show that it outperforms previous approaches in terms of both accuracy and diversity, thereby yielding a new state-of-the-art in activity anticipation.

1 Introduction

Activity anticipation has drawn considerable recent attention and has evolved from relatively simple next frame prediction to more challenging asynchronous sequences of activities and times that occur further into the future [34,35]. One particular characteristic of activities, uncertainty, is also gaining consideration [37,1]. For example, given certain starting procedures of preparing a meal in a kitchen, initial actions *take_egg, butter_pan* can be followed by *crack_egg, fry_egg* for a fried egg breakfast, or alternatively *spoon_flour, pour_milk, etc.* for a pancake breakfast, as well as many other possibilities as in Fig. 1. Indeed, not only the activities, but also the durations can vary (*e.g.* cooking time differs between *soft* and *hard* boiled eggs). This phenomenon, referred to as diversity in prediction, is one of the core abilities of the human prediction system, while it is much less well captured in contemporary computer vision.

Recent efforts on increasing diversity have concentrated on generative models. GANs [13] are especially notable for generating near realistic (*i.e.* high quality) samples in the continuous domain [57,26,3,27], To date, however, activity anticipation has barely benefited from these developments. The main issue lies in the discrepancy between generating in continuous and discrete spaces, with action labels most naturally being discrete (*e.g.* one-hot representations) while times typically are taken as continuous (*e.g.* real valued numbers). This innate disparity has led to different learning and inference strategies as well as difficulties in encompassing them within adversarial learning.

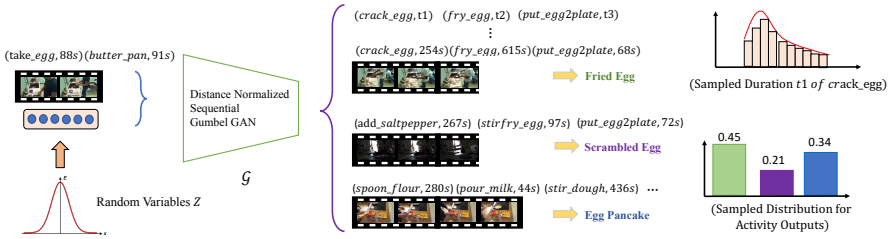


Fig. 1: Proposed Activity Anticipation Process Overview. An initial observation of an activity, as a discrete sequence of actions and times (*e.g.* take_egg, 88s, *etc.*) and a random variable, z, input to a sequence generator, $\mathcal{G}$. The generator combines adversarial learning with Gumbel sampling and normalized distance regularization so that sampled outputs are both diverse and realistic.

Moreover, the general task of discrete data generation confronts challenges: Its typical procedure involves data sampling, which is not differentiable and thus standard gradient-based learning techniques cannot be applied directly in these settings [39]. To bypass this limitation, recent efforts have introduced reinforcement learning (RL) [55] to estimate a policy gradient and *Gumbel-Softmax relaxation* [16] that performs reparameterization to allow gradient flow. However, both of these approaches introduce their own limitations: It is unclear how RL based approaches affect diversity; reparameterizing incurs a trade-off between quality and diversity, which prevents generating both realistic and diverse (*i.e.* multimodal) outputs. As noted by others [31,53], insufficient diversity mainly derives from typical mode-collapse issues in GANs[13].

In response to the above challenges, we focus on developing a practical and effective approach that applies a GAN on sequences of action-time pairs, while alleviating mode-collapse in adversarial discrete sequence generation. To achieve this result, we begin by revisiting the representation of time to cast it as a discete variable and relate this approach to alternatives in the recent literature. Notably, standard activity anticipation datasets (*e.g.* [6,47,22]) provide times in integer units; so, our discrete temporal representation is more consistent with that format than a continuous representation and also allows us to unify inference of activities and their times. Next, we merge the *Normalized Distance Regularizer* [53,31], which serves to combat mode-collapse, with *Gumbel-Softmax* based discrete sequential GANs, which provides realistic instance generation. We then formulate activity anticipation as a conditional adversarial generation problem in the joint action-time discrete domain. Specifically, partial video information in the format of discrete sequences of categorical action labels and their temporal durations are taken as input to generate the remainder of the activity in the same format, *i.e.* sequential action labels and their temporal durations. Empirical evaluation shows that our approach yields new state-of-the-art performance on three popular activity anticipation datasets. Figure 1 provides an overview. Our implementation is available at our project page.

2 Related research

With the previous success of action recognitions, *e.g.* [44,52,9,4], some research has refocused on two natural extensions, action prediction and activity anticipation. The former refers to recognizing actions as early as possible, while the latter

to predicting subsequent future actions. In either case, much work relied on visual feature representation learning from raw videos [20,21,43,56,46,45,51,11]. Alternatively, to better capture future state-space dynamics in complex scenes (*e.g.* pedestrian trajectories or sports player motions), higher-level abstractions (*e.g.* point tracks) have shown to be more practical than raw inputs [54,29,48,30,38].

Indeed, recent approaches that depend on high-level abstraction from videos (*e.g.* semantic labels and times) have also shown promising results for both near and long term activity as well as caption anticipation [35,34,2,17]. Along similar lines, but with a focus on uncertainty modeling, work has been developed that learned a parameterized variational temporal point process to capture the distributions of activity categories and starting times of single immediatelly following actions [37]. Other work that focused on uncertaintly modeling used beam search to make final selection from a pool of action candidates [42] as well as multi-label learning [10]. Yet other work predicted all subsequent activities and corresponding durations in a stochastic manner [1]. Our research is most similar to the final example in predicting all subsequent actions and durations with a special focus on predictions that are both realistic (high quality) and diverse.

During learning, action labels and times are usually separated [35,34,2,17,1], *e.g.* action labels are taken as classifications and optimized under cross-entropy (CE), while times as real-valued variables and optimized with mean squre error (MSE). CE suffers from overly high resemblance to dominant groundtruth, while suppressing other reasonable possibilities [2,5] and MSE is known for producing blurred outcomes [36]. Recently, superior results have been demonstrated by taking time as a discrete (integer) input to learn exponential-family distributions that are optimized with negative log-likelihood, but still detached from label optimizations [37]. Other work also has shown that framing time as a discrete variable can yield strong results[32,28]. Our approach extends such work by taking input time as a discrete variable, but outputs a softmax discrete random variable and further integrates action and time learning in an adversarial framework to yield naturally unified predictions.

Arguably, the major obstacle to further progress in exploiting categorical models of activity anticipation is the inadequacy of existing models for discrete sequence generation. Still, some progress has been made. Two groups ([16,33]) simultaneously proposed softmax relaxation for discrete random variables; however, they only evaluated on simple datasets under variational inference. Recent efforts in text generation use adversarial generative networks by estimating gradients via a reinforcement policy; however, such approaches are known for training difficulties and high gradient variance [55]. Other work managed to obtain success with the adversarial Gumbel-trick [39], which, however, incurrs a tradeoff between quality and diversity; see Supplemental Material for a detailed example.

Different from previous work, we successfully model activity anticipation in an adversarial generative fashion in the discrete domain. We give a solution for generating outputs having both quality and diversity under Gumbel relaxation via incorporation of a distance regularizer. The upgraded relaxation framework is applied to conditional discrete sequence generation. To the best of our knowledge,

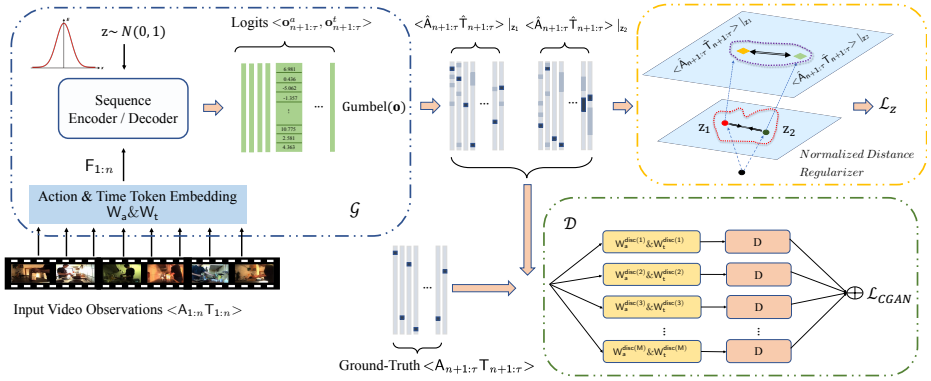


Fig. 2: Depiction of our diversity improved discrete sequential GAN architecture. For both training and inference, input action and time duration token sequences $<\mathsf{A}_{1:n}\mathsf{T}_{1:n}>$ are embedded via learnable matrices $\mathsf{W_a}$ and $\mathsf{W_t}$, (1), to generate a compact feature representation, $\mathsf{F}_{1:n}$. $\mathsf{F}_{1:n}$ is processed iteratively along with latent noise signals, z, by an encoder (2) and decoder(3) (RNN or LSTM) along with a linear mapping, (4), to produce logits $<\mathbf{o}^a_{n+1:\tau}, \mathbf{o}^t_{n+1:\tau}>$. At inference time (not shown), the logits are simply sampled with argmax, (5), to produce predicted activity sequences. At training time (shown), Gumbel sampling, (10), is performed to facilitate gradient-based learning. Pairs of samples, $<\hat{\mathsf{A}}_{n+1:\tau}\hat{\mathsf{T}}_{n+1:\tau}>|_{z_1}$ and $<\hat{\mathsf{A}}_{n+1:\tau}\hat{\mathsf{T}}_{n+1:\tau}>|_{z_2}$, produced with different latents, z_1 and z_2, are compared under a normalized distance regularizer, (12), to define a loss, $\mathcal{L}_z$, that encourges a generator capable of diverse ouputs. In complement, a sampled output is adversarially compared, (13), to groundtruth, $<\mathsf{A}_{n+1:\tau}\mathsf{T}_{n+1:\tau}>$, by a discriminator, $\mathcal{D}$, under multiple embeddings, $\mathsf{W}_\mathsf{a}^{\mathsf{Disc(i)}}, \mathsf{W}_\mathsf{t}^{\mathsf{Disc(i)}}$, to define an additional loss, $\mathcal{L}_{CGAN}$, that encourages a generator capable of realistic outputs.

ours is the first study that explores a treatment to mode collapse in the discrete sequence generation domain, including an integrated discrete representation of action labels and times and use of conditional GANs for activity anticipation.

3 Technical approach

3.1 Overview

We seek to anticipate what will happen next from an observed initial part of a video, *c.f.* [2,37,17]. We are interested in not just anticipating a single action that immediately follows the observed video, but potentially an entire sequence of subsequent actions. Specifically, the input contains a sequence of n observed action tokens $\mathsf{A}_{1:n}= (\mathbf{a}_1, \ldots, \mathbf{a}_n)$, with $\mathbf{a}_i$ the i^{th} token in the sequence. Actions are taken to belong to an action vocabulary of size V, *i.e.* $\mathbf{a}_i \in \mathbb{R}^V$, and each $\mathbf{a}_i$ is a one-hot vector action encoding. Correponding to each action sequence, $\mathsf{A}_{1:n}$, is a time sequence, $\mathsf{T}_{1:n}= (\mathbf{t}_1, \ldots, \mathbf{t}_n)$ with $\mathbf{t}_i$ the temporal length (*e.g.* in units of seconds) of the i^{th} action belonging to a duration vocabulary of size H, $\mathbf{t}_i \in \mathbb{R}^H$. Each $\mathbf{t}_i$ is a one-hot vector encoding of the duration of action at position i in the sequence, taken from the vocabulary of temporal durations.

Our temporal representation is motivated by a desire to treat encoding, decoding and adversarial learning in a consistent fashion across input, output and groundtruth and will be shown to yield state-of-the-art results. In all three of the considered standard datasets, the time data format (*e.g.* seconds) is given as discrete. Previous work [1,17,37], using MLPs for time decoding of future activities causes difficulty in learning, given discrete groundtruth. Also, recursive predictions that rely on preceding results suffer in such settings, as only dis-

crete values are in the training data. We match our representation to the data by enforcing one-hots for prediction. It has been shown beneficial to treat such formatting as categorical under adversarial learing in allied domains [5,24,39].

We formulate the task as: Given such a partial sequence as an ordered pair, $<\mathsf{A}_{1:n}, \mathsf{T}_{1:n}>$, we produce a distribution over subsequent action-temporal duration pairings, $<\mathsf{A}_{n+1:\tau}, \mathsf{T}_{n+1:\tau}>$, up to and including time τ. Our activity anticipation pipeline has three parts: 1) an input sequential token embedding module, 2) an output sequential generator, $\mathcal{G}$, and 3) a ConvNet based discriminator, $\mathcal{D}$, for adversarial learning. We describe each separately in the following.

3.2 Dual token embedding

As widely discussed in the NLP and image/video captioning literatures, the expressive power of raw discrete tokens can be greatly enhanced through dictionary embeddings in higher dimensional continuous feature spaces, *c.f.* [2,17,37]. To this end, each entry of the initial pairing of action and temporal tokens $\mathbf{a}_i$ and $\mathbf{t}_i$ are projected to higher dimension continuous spaces, $\Re^a$ and $\Re^t$, resp., via dictionary mapping matrices W_a and W_t. Here, W_a, of dimension $\Re^a \times V$, is the mapping for actions, while W_t, of dimension $\Re^t \times H$, is the mapping for times. The embedding matrices are constructed such that their j^{th} columns are the mappings for actions and times indicated by the one-hot vector encodings of actions and times. (Note that for a given observation pair, $<\mathbf{a}_i, \mathbf{t}_i>$, j generally is different for the action and time.) Our overall embeddings are then given as

$$\mathbf{f}_i = Cat\big[\mathsf{W}_\mathsf{a}\mathbf{a}_i, \mathsf{W}_\mathsf{t}\mathbf{t}_i\big] \tag{1}$$

with $\mathbf{f}_i$ associating action and temporal features at the i^{th} index via concatenation of their respective embedding vectors. Analogous to the input sequences, $\mathsf{A}_{1:n}$ and $\mathsf{T}_{1:n}$, we define embedded sequences, $\mathsf{F}_{1:n}= (\mathbf{f}_1, \ldots, \mathbf{f}_n)$.

3.3 Sequence generation

For computing outputs, $<\mathsf{A}_{n+1:\tau}, \mathsf{T}_{n+1:\tau}>$, given a sequence of input feature embeddings, $\mathsf{F}_{1:n}$, we use the *seq2seq* generator [49]. To lend insight into the generality of our approach, we consider two different backbones, RNN and LSTM, as instantiating mechanisms for the generator and results for both are reported in Sec. 4. While this design choice might be replaced by additional alternatives (*e.g.* a temporal convolutional network (TCN) [40,7] or relational memory [41,39]), RNN and LSTM are general sequence learning structures also adopted by our main comparison approaches [2,1]. In the following, we make use of a general notation such that *Sequence* stands for either RNN or LSTM.

Sequence generation proceeds by sequentially encoding and decoding the embedded features, followed by mapping to logits and sampling. More specifically, the components of the input feature sequence, $(\mathbf{f}_1, \ldots, \mathbf{f}_n)$, are iteratively injected into an encoder, $Sequence^{enc}$, to obtain a fixed dimensional representation $\mathbf{v}$, given by the last hidden state of $Sequence^{enc}$, according to

$$\mathbf{h}_k^{enc} = Sequence^{enc}(\mathbf{f}_k, \mathbf{h}_{k-1}^{enc}), 1 < k < n \quad \text{and} \quad \mathbf{v} = \mathbf{h}_n^{enc}, \tag{2}$$

$\mathbf{h}_k^{enc}$ the k^{th} hidden encoder state and the initial hidden state, $\mathbf{h}_0$, randomly set.

Next, a decoding sequence model, $Sequence^{dec}$, whose initial hidden state is set to $\mathbf{v}$ and first input variable to the last observation $\mathbf{f}_n$, is used for producing output hidden state sequence $(\mathbf{h}^{dec}_{n+1}, \ldots, \mathbf{h}^{dec}_{\tau})$ according to

$$\mathbf{h}^{dec}_{k+1} = Sequence^{dec}(\mathbf{f}_k, \mathbf{h}^{dec}_k), n < k < \tau - 1. \tag{3}$$

Then, a pair of linear transformations defined by matrices $\mathsf{W}^{\phi}_{\mathsf{o}}$ and vectors $\mathbf{b}^{\phi}_o$, $\phi \in \{a, t\}$, maps hidden states, $\mathbf{h}^{dec}_k$, to logits according to

$$\left.\begin{aligned} \mathbf{o}^a_k &= \mathsf{W}^{\mathsf{a}}_{\mathsf{o}}\mathbf{h}^{dec}_k + \mathbf{b}^a_o \\ \mathbf{o}^t_k &= \mathsf{W}^{\mathsf{t}}_{\mathsf{o}}\mathbf{h}^{dec}_k + \mathbf{b}^t_o \end{aligned}\right\} \quad n+1 \leq k \leq \tau, \tag{4}$$

where $\mathbf{o}^a_k$ and $\mathbf{o}^t_k$ are vectors representing the logits for each entry of action or time vocabulary, *i.e.* they are of dimensions $\Re^a \times V$ and $\Re^t \times H$, resp.

Finally, the logits, (4), are sampled in one of two ways, depending on whether the system is operating in inference or training mode. At inference time, traditional discrete data sampling is used to select the most likely probability index, which is then reformatted as one-hot vectors according to

$$\left.\begin{aligned} \mathbf{a}_k &= one\_hot\left(\underset{i}{\operatorname{argmax}}(\mathbf{o}^a_k[i])\right) \\ \mathbf{t}_k &= one\_hot\left(\underset{i}{\operatorname{argmax}}(\mathbf{o}^t_k[i])\right) \end{aligned}\right\} \quad n+1 \leq k \leq \tau, \tag{5}$$

with $\mathbf{o}^a_k[i]$ and $\mathbf{o}^t_k[i]$ selecting the i^{th} element of $\mathbf{o}^a_k$ and $\mathbf{o}^t_k$, resp.

Unfortunately, the argmax operator incurs zero derivative with respect to parameters of operations coming before it, *i.e.* in the present context those of the embedding matrices and generator, which interferes with gradient-based training. Therefore, to obtain differentiable one-hot vectors from logits $\mathbf{o}_k$ during training, we adopt the Gumbel-Softmax relaxation technique [16,33] that mimics one-hot vectors from categorical distributions. In particular, we replace (5) with Gumbel sampling, (10), which yields the output discrete tokens according to

$$\left.\begin{aligned} \mathbf{a}_k[i] &= \mathsf{Gumbel}(\mathbf{o}^a_k[i]) \\ \mathbf{t}_k[i] &= \mathsf{Gumbel}(\mathbf{o}^t_k[i]) \end{aligned}\right\} \quad n+1 \leq k \leq \tau. \tag{6}$$

For both sampling techniques (5) and (6), for each iteration on k, we use

$$\mathbf{f}_k = Cat\big[\mathsf{W}_{\mathsf{a}}\mathbf{a}_k, \mathsf{W}_{\mathsf{t}}\mathbf{t}_k\big], n+1 \leq k \leq \tau \tag{7}$$

in the encoding, (2), to reinitialize subsequent decoding (3), logits mapping (4), and sampling (5) or (6), to give the next generation of action/time tokens. Having produced $\mathbf{a}_k$ and $\mathbf{t}_k$ for $n+1 \leq k \leq \tau$ via (6), we define $\mathsf{A}_{n+1:\tau} = (\mathbf{a}_{n+1}, \ldots, \mathbf{a}_{\tau})$, $\mathsf{T}_{n+1:\tau} = (\mathbf{t}_{n+1}, \ldots, \mathbf{t}_{\tau})$ and ultimately $<\mathsf{A}_{n+1:\tau}, \mathsf{T}_{n+1:\tau}>$ as output.

Overall, the generating process, $\mathcal{G}$, of our approach entails sequential application of the embedding (1), encoding (2), decoding (3), logits mapping (4) and sampling, (5) or (6), to each element of the input, $<\mathsf{A}_{1:n}, \mathsf{T}_{1:n}>$, formalized as

$$<\mathsf{A}_{n+1:\tau}, \mathsf{T}_{n+1:\tau}> = \mathcal{G}(<\mathsf{A}_{1:n}, \mathsf{T}_{1:n}>, z; \theta_g) \tag{8}$$

where θ_g includes learnable parameters of $Sequence^{enc}$, $Sequence^{dec}$, $\mathsf{W}_{\phi}, \mathsf{W}^{\phi}_{\mathsf{o}}, \mathbf{b}^{\phi}_o$, $\phi \in \{a, t\}$, and z is a randomized input that encourages diversity during sam-

pling; see Sec 3.4. From a probabilistic view, the overall sequence generation procedure can be seen to operate as a Markov process,

$$p(\mathsf{A}_{n+1:\tau}, \mathsf{T}_{n+1:\tau} | \mathsf{A}_{1:n}, \mathsf{T}_{1:n}) = \prod_{\mathsf{k=n+1}}^{\tau} \mathsf{p}(\mathbf{a_k}, \mathbf{t_k} | \mathbf{v}, \mathbf{f_n}, \mathsf{h_{n+1}}, ..., \mathsf{h_{k-1}}). \tag{9}$$

3.4 Adversarial learning and the discriminator

Gumbel sampling During training, sampling of the logits is formulated as Gumbel-noise reparameterization, referred to as the *Gumbel-Max trick* [14,33], to allow gradients to flow for end-to-end optimization according to

$$\mathsf{Gumbel}(\mathbf{o}_k[i]) \equiv \frac{\exp((\log(\mathbf{o}_k[i] + \mathbf{g}_k[i])\,\alpha)}{\sum_{j=1}^{m} \exp((\log(\mathbf{o}_k[j]) + \mathbf{g}_k[j])\,\alpha)}, \tag{10}$$

with m the dimension of $\mathbf{o}_k$ and each element of $\mathbf{g}_k$ drawn from the $i.i.d.$ standard Gumbel distribution [14], $-\log(-\log U_k)$, with U_k drawn from $unif(0, 1)$.

In the definition of Gumbel, (10), α is a tunable parameter, called the *temperature*, that controls the similarity of the approximated one-hot vector, (6) to the actual one-hot vector, (5). Intuitively, smaller α provides more accurate outputs for the discriminator, thereby leading to better sample quality. However, the associated drawback is high gradient variance causing mode-collapse [39]. To combat this situation, one can adopt a temperature annealing strategy to encourage diversity [33,16], which is heuristic in terms of how much and when the temperature should be annealed. Instead, we make use of the recently proposed *Normalized Distance Regularizer* [53,31] to augment (10), as detailed next.

Normalized distance regularizer To get diversity in generated sequences, we use a *Normalized Distance Regularizer* [53,31]. This regularizer operates via pairwise distances between various outputs from the same input according to

$$\max_{\mathcal{G}} \mathcal{L}_z(\mathcal{G}) = \mathbb{E}_{z_1, z_2} \left[\min \left(\frac{||\mathcal{G}(\mathsf{Z}, z_1) - \mathcal{G}(\mathsf{Z}, z_2)||}{||z_1 - z_2||} \right) \right], \tag{11}$$

where Z is input (*e.g.* in our case, an observed action-time sequence pair, $<\mathsf{A}_{1:n}, \mathsf{T}_{1:n}>$), $\mathbb{E}[\cdot]$ is the expectation operator, $||\cdot||$ is a norm and $z_i \sim N(0, 1)$ is a random latent variable that converts the deterministic model into a stochastic one. The normalized distance regularizer, (11), operates such that the denominator, $||z_1 - z_2||$, encourages the two random codes to be close in latent space, while the numerator, $||\mathcal{G}(\mathsf{Z}, z_1) - \mathcal{G}(\mathsf{Z}, z_2)||$, encourages the outputs to be distant in the output space. The intuition is to encourage $\mathcal{G}$ to visit more important modes given an input variable, but with minimum traveling effort in the sampling space. In this way, distinctive outcomes are circumscribed within a tighter latent space, so that the model is less likely to ignore some modes [53,31].

In application to our case, the overall generated sequence $<\mathsf{A}_{n+1:\tau}, \mathsf{T}_{n+1:\tau}> = \mathcal{G}(<\mathsf{A}_{1:n}, \mathsf{T}_{1:n}>, z)$, is used to calculate the l_1 distance between samples,

$$\mathcal{L}_z(\mathcal{G}) = \quad \mathbb{E}_{z_1, z_2} \left[\frac{||\mathcal{G}(<\mathsf{A_{1:n}}, \mathsf{T_{1:n}}>, z_1) - \mathcal{G}(<\mathsf{A_{1:n}}, \mathsf{T_{1:n}}>, z_2)||_{l_1}}{||z_1 - z_2||} \right], \tag{12}$$

where $||\cdot||_{l_1}$ denotes use of the l_1 norm; explicit dependence of $\mathcal{G}$ on θ is suppressed for conciseness. Details of how (12) is realized are in the Supplement.

It appears that our solution to preserving both quality and diversity in discrete sequence generation is novel. Beyond evidence for the effectiveness of the approach from Sec. 4 experiments, the Supplement has a detailed toy example.

Multi-embedding discriminator To train our generator, $\mathcal{G}$, adversarially, we refer its output to a discriminator, $\mathcal{D}$. Successful examples of deep discriminators for sequence generation include DNNs, ConvNets and RNNs [50,18,24], amongst which incorporation of a ConvNet classifier with an ensemble of multiple embeddings has achieved top performance [39]. Notably that approach has shown an ability to dispose of discriminator pretraining. We adopt that approach.

Given a generated output sequence, $<\hat{\mathsf{A}}_{n+1:\tau}, \hat{\mathsf{T}}_{n+1:\tau}>$, and real (*i.e.* ground truth) sequence, $<\mathsf{A}_{n+1:\tau}, \mathsf{T}_{n+1:\tau}>$, we embed the elements of the sequences analogous to the embeddings of Sec. 3.2, and execute our discriminator, $\mathcal{D}$. Let $\mathsf{W}_{\mathsf{a}}^{\mathsf{Disc}}$ and $\mathsf{W}_{\mathsf{t}}^{\mathsf{Disc}}$, be discriminator embedding matrices, distinct from, but analogous to the embedding matrices used earlier, (1). Then, the process unfolds in three steps. First, for each pair $<\mathbf{a}_i, \mathbf{t}_i>, i \in \{n+1, \tau\}$, and $<\hat{\mathbf{a}}_i, \hat{\mathbf{t}}_i>, i \in \{n+1, \tau\}$ we produce embedded vectors $\mathbf{f}_i^{\mathsf{Disc}}$ and $\hat{\mathbf{f}}_i^{\mathsf{Disc}}$ by replacing W_{a} and W_{t} with $\mathsf{W}_{\mathsf{a}}^{\mathsf{Disc}}$ and $\mathsf{W}_{\mathsf{t}}^{\mathsf{Disc}}$, resp. in (1). Second, the complete sets of produced embeddings $\mathbf{f}_i^{\mathsf{Disc}}$ and $\hat{\mathbf{f}}_i^{\mathsf{Disc}}$ are converted into matrices, $\mathsf{F}_{\mathsf{n+1:\tau}}^{\mathsf{Disc}}$ and $\hat{\mathsf{F}}_{n+1:\tau}^{\mathsf{Disc}}$, whose i^{th} columns are $\mathbf{f}_i^{\mathsf{Disc}}$ and $\hat{\mathbf{f}}_i^{\mathsf{Disc}}$, resp. Third, the resulting $\mathsf{F}_{\mathsf{n+1,\tau}}$ and $\hat{\mathsf{F}}_{n+1:\tau}$ are processed akin to 2D images with a discriminator ConvNet, $\mathcal{D}$, consisting of multiple convolutional, nonlinear activation and max-pooling layers with a fully connected layer at the top to produce a label, *i.e.* 1 *vs.* 0 for real *vs.* generated.

Finally, to account for the desired multiple embeddings, the entire three step process is repeated M times with distinct embeddings, $\mathsf{W}_{\mathsf{a}}^{\mathsf{Disc}(i)}$ and $\mathsf{W}_{\mathsf{t}}^{\mathsf{Disc}(i)}$, $i \in \{1, \ldots, M\}$, and the final adversarial loss from $\mathcal{D}$ comes by averaging across the results of those embeddings according to

$$l_{\mathcal{D}} = \frac{1}{M} \sum_{i=1}^{M} \log(1 - \mathcal{D}(\hat{\mathsf{F}}_{n+1,\tau}^{\mathsf{Disc}(i)}, 0; \theta_d)) + \log(\mathcal{D}(\mathsf{F}_{\mathsf{n+1,\tau}}^{\mathsf{Disc}(i)}, 1; \theta_{\mathsf{d}})), \tag{13}$$

with θ_d the learnable parameters of the M matrix embeddings and $\mathcal{D}$.

Combined objective Our learning goal is to generate samples that are both realistic (high quality) and diverse. To this end, we merge the standard adversarial learning formula (for quality)

$$\mathcal{L}_{CGAN}(\mathcal{G}, \mathcal{D}; \theta_g, \theta_d) = \mathbb{E}_y[\log \mathcal{D}(y, 1; \theta_d)] + \mathbb{E}_{x,z}[\log(1 - \mathcal{D}(\mathcal{G}(x, z; \theta_g), 0; \theta_d))], \tag{14}$$

which in our case specializes to $l_{\mathcal{D}}$, (13), with the normalized distance regularizer, (12), for diversity, to consider

$$\min_{\mathcal{G}} \max_{\mathcal{D}} \lambda_1 \mathcal{L}_{CGAN}(\mathcal{G}, \mathcal{D}; \theta_g, \theta_d) - \lambda_2 \mathcal{L}_z(\mathcal{G}, \theta_g), \tag{15}$$

which we optimize over θ_g and θ_d according to methods documented under implementation details. The overall learning process is summarized in Fig. 2.

4 Empirical evaluation

4.1 Datasets

We evaluate on three standard datasets.

Breakfast Dataset contains 1,712 videos of 52 actors making breakfast using 48 fine-grained actions. We follow the data processing strategy used elsewhere for ConvNet modeling [2], which generates 4 sets of training examples by using the first 10%, 20%, 30% and 50% of the video, resp., as observation and the rest of the video as ground-truth for prediction evaluation. This procedure yields 7305 training samples and 253 testing samples. We report average accuracy across four splits, as elsewhere [23].

50Salads Dataset has 50 videos of 25 subjects, each making 2 salads using 17 fine grained actions. The same processing used for the Breakfast dataset is adopted here, which leads to 160 training samples and 40 testing samples. We perform 5-fold cross-validation for evaluation, using previous splits [25].

Epic-Kitchens Dataset contains 1^{st}-person videos of 32 actors in 32 kitchens performing unscripted daily activities. There are 272 training videos recorded from 28 subjects, with the rest used for testing. In total there are 128 actions. We follow the preprocessing strategy used previously that randomly split the training video into 7 subsets, each containing 4 subjects [17]. The final result is averaged across all subsets. For each train and test input, an arbitrary clip is sampled by applying temporal windows of 30 seconds and the following 60 seconds is taken as groundtruth prediction.

4.2 Implementation details

Groundtruth time format We conform to previous temporal formats [2,37]. For Breakfast and 50Salads, we use temporal intervals in units of seconds as observed in the original videos. For example, in the minutes long *egg_cake* video from Breakfast, the 3^{rd} action in the vocabulary is *cracking_egg* and occupies 15s, which we correspondingly represent as (3, 15). Accordingly, the column entry size of the time embedding matrix, $\mathsf{W_t}$, is set to the maximum action length across the entire dataset, which is 5791s for Breakfast and 4149s for 50Salads, by our counting. For Epic-Kitchens, time units are also in seconds. Random temporal crops of 90s are extracted from the videos, with the first 30s taken as observation and the remaining 60 for prediction. So, the column size of $\mathsf{W_t}$ is 90, *c.f.* [17]. We set the output length to 25 and 20 for Breakfast and 50Salads, resp., based on the maximum sample length in each dataset and pad zeros for shorter samples. The generated length is 60 for Epic-Kitchen by default, *c.f.* [17].

Network configurations We set the embedding dimension for input actions and times to be 32 while the hidden states of $Sequence^{enc}$ and $Sequence^{dec}$ to have dimension 128. The number of embeddings for the discriminator is M=64, each of dimension 32. For noise, z, we sample a 32-dimensional random latent variable from the $N(0, 1)$ distribution, *c.f.* [16] and concatenate it to the input at each time step of the generator, $\mathcal{G}$, *c.f.* [53]. For the ConvNet based discriminator, $\mathcal{D}$, four 2D convolutional layers with 2×2 kernels are followed by ReLU [12] and max-pooling with factor of 2 subsampling. A fully connected layer is appended

Observation	20%				30%			
Prediction	10%	20%	30%	50%	10%	20%	30%	50%
RNN [2]	0.6035	0.5044	0.4528	0.4042	0.6145	0.5025	0.4490	0.4175
CNN [2]	0.5797	0.4912	0.4403	0.3926	0.6032	0.5014	0.4518	0.4051
TOS-Dense [17]	0.6446	0.5627	0.5015	0.4399	0.6595	0.5594	0.4914	0.4423
RNN-HMM (**Avg**) [1]	0.5039	0.4171	0.3779	0.3278	0.5125	0.4294	0.3833	0.3307
RNN-HMM (**Max**) [1]	0.7884	0.7284	0.6629	0.6345	0.8200	0.7283	0.6913	0.6239
Ours-RNN (**Avg**)	0.7101	0.6200	0.5421	0.4383	0.7304	0.7053	0.6257	0.5119
Ours-RNN (**Max**)	0.8076	0.7010	0.6649	0.6283	0.8210	0.7539	0.7134	0.6251
Ours-LSTM (**Avg**)	**0.7222**	**0.6240**	**0.5622**	**0.4595**	**0.7414**	**0.7132**	**0.6530**	**0.5238**
Ours-LSTM (**Max**)	0.8208	0.7059	0.6851	0.6406	0.8336	0.7685	0.7213	0.6406

Table 1: Breakfast Dataset Protocol 1 results with dense anticipation mean over classes (MoC) accuracy. Avg stands for averaged results across 16 samplings, while Max stands for taking the best result among 16 samples. Best Avg is highlighted.

on top for 1 *vs*. 0 classification for groundtruth/real *vs*. generated, resp. These configurations are applied identically for both RNN and LSTM backbones.

Training Sequential GANs suffer from training difficulties that usually necessitates pretraining with maximum likelihood estimation (MLE) for both generator and discriminator [55,8,15]. Recently, RelGAN [39] successfully avoided discriminator pretraining, while still relying on it for the generator under the Gumbel GAN structure. We also found pretraining unnecessary for our discriminator, but still begin with generator pretraining using MLE. Both pretraining and adversarial training employ the Adam optimizer [19]. Mini-batch-Stochastic Gradient Descent is used with a learning rate of $1e^{-2}$ for pre-training and $1e^{-4}$ for adversarial training, both with exponential decay of $1e^{-5}$. Empirically, we set λ_1, λ_2 to a ratio of 1:1. For both training and testing we input groundtruth tokens (action and time labels), with subsequent groundtruth beyond the input (*i.e*. prediction) used for training only, as with the compared results [2,17,37].

4.3 Anticipation results across datasets

Breakfast dataset protocol 1 When evaluating on this dataset, we first conduct experiments under the setting where partial observation video is obtained by arbitrarily cutting from the entire sequence [2,17,1]. Results are reported for combinations of observation (20% and 30%) *vs*. prediction (10%, 20%, 30% and 50%). Table 1 shows comparisons. As our model captures uncertainty via use of randomized latents, we report two evaluation metrics, average and maximum (*i.e*. mean or max accuracy across multiple samples, 16 for results shown).

It is seen that our approach using RNN or LSTM outperforms the alternatives in almost all settings by an average of 5−8% accuracy under the average metric, with the exception being the most extreme case, *i.e*. 50% prediction from 20% observation, where TOS-Dense [17] performs slightly better (by merely 0.16%) than our RNN backbone; however, our LSTM backbone still wins by 0.96%. Given more information to start with, as in 30% observation, both of our models uniformly surpass the others. The biggest difference between our instantiations reside in remote predictions, *i.e*. 30% and 50%, affirming that LSTM generally is better than RNN in long-term performance [17]

Compared with another stochastic approach, RNN-HMM [1], our RNN model achieves similar performance on the **Max** metric while notably excelling on the **Avg** metric. Under both metrics our LSTM model is the top performer. Arguably, our model owns a similar base architecture as [2] and is less compli-

Model	Loss	stoch. var.	accuracy	MAE	LL
TD-LSTM [37]	CE + MSE	-	53.64	173.76	-
APP-LSTM [37]	CE + NLL	-	61.39	152.17	-6.668
RNN-HMM [1]	CE + MSE	✓	57.80	-	-
APP-VAE− [37]	NLL+KL	✓	27.09	270.75	$\geq$ -9.427
APP-VAE+ [37]	NLL+KL	✓	62.20	142.65	$\geq$ -5.944
Ours-RNN	Adv+NormDist	✓	**68.46**	**87.58**	$\geq$ **-4.851**
Ours-LSTM	Adv+NormDist	✓	**68.96**	**87.07**	$\geq$ **-4.836**

Table 2: Breakfast Dataset Protocol 2 results. Note that larger accuracy is better, while smaller MAE is better. For LL, smaller absolute value is better. APP-VAE+ and APP-VAE− refer to APP-VAE with and without a learned prior, resp.

cated than TOS-Dense [17], which introduced skip-connection and attention, yet achieves better results. We attribute this pattern to two reasons: 1) Our approach avoids error propagation coming with iterative long-term generation [2,1], where anticipation errors in early stages are propagated to the end. 2) Our adversarial learning focuses on the quality of overall generated sequences, rather than any single prediction. The discriminator pushes the generator to produce outputs that are more realistic as a whole.

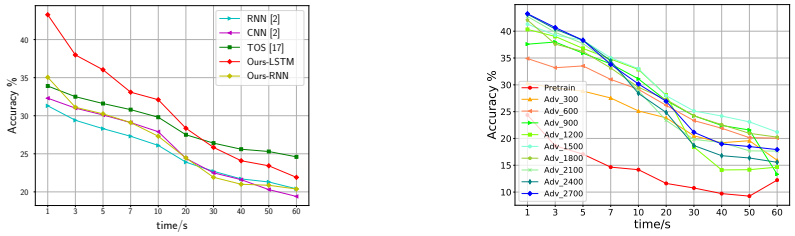
Breakfast dataset protocol 2 To further demonstrate our strength in capturing uncertainty, we present an experiment using the protocol from Action Point Process (APP-VAE) [37], which aims to capture stochastics of single next action anticipation, but with a variational inference structure. In this scenario, the input sequence is obtained by clipping the whole video at the exact end point of random actions. Log-Likelihood (LL) is used as a metric for measuring the approximated posterior via standard importance sampling *c.f.* [37]. Our method is accommodated to this evaluation by selecting the immediate next action from the generated sequence. To report a lower bound LL, we chose the worst score (largest absolute value) from multiple runs. Details of the LL evaluation are presented in the Supplemental Material. Results are shown in Table 2.

Both our models achieves better LL compared to the previous best of -5.944, with our LSTM at -4.836 slightly outperforming our RNN, after averaging inference results from test sets. Since this protocol focuses on a single next action, results from our two backbones are close. We also calculate the Mean Classification Accuracy over the entire anticipated sequence for action category performance and find our results outperform APP-VAE by ≈7%. For temporal duration estimation, we use Mean Absolute Error (MAE) as the evaluation metric and find that our approach reduces the time length estimation error by ≈40%.

Observation	20%				30%			
Prediction	10%	20%	30%	50%	10%	20%	30%	50%
RNN [2]	0.4230	0.3119	0.2522	0.1682	0.4419	0.2951	0.1996	0.1038
CNN [2]	0.3608	0.2762	0.2143	0.1548	0.3736	0.2478	0.2078	0.1405
TOS-Dense [17]	0.4512	0.3323	0.2759	0.1727	**0.4640**	0.3480	0.2524	0.1384
RNN-HMM (**Avg**) [1]	0.3495	0.2805	0.2408	0.1541	0.3315	0.2465	0.1884	0.1434
RNN-HMM (**Max**) [1]	0.7489	0.5875	0.4607	0.3571	0.6739	0.5237	0.4673	0.3664
Ours-RNN (**Avg**)	0.4571	0.3517	0.3182	0.2095	0.4597	0.3591	0.3073	0.1746
Ours-RNN (**Max**)	0.4903	0.3998	0.3596	0.2537	0.4926	0.4582	0.3572	0.2646
Ours-LSTM (**Avg**)	**0.4663**	**0.3562**	**0.3191**	**0.2137**	0.4613	**0.3637**	**0.3310**	**0.1945**
Ours-LSTM (**Max**)	0.5150	0.3845	0.3606	0.2762	0.5079	0.4754	0.3783	0.2908

Table 3: Results for the 50Salads dataset with dense anticipation mean over classes (MoC) accuracy.

50 Salads dataset We adopt the same experimental design as for Breakfast Protocol 1; results are in Table 3. As in the previous experiment, both our



(a) *Epic-Kitchen dataset* (b) *Ablation on Epic-Kitchen dataset*

Fig. 3: Comparison table for the Epic-Kitchen dataset (a). Ablation study on Ours-LSTM for the Epic-Kitchen dataset. Adv_$*$ indicates $*$ adversarial training epochs (b).

models achieve better average performance than the alternatives for most cases, but slightly worse at the 30% observation *vs*. 10% prediction ratio (LSTM worse by merely 0.27%). When it comes to 30% and 50% prediction scales, our models hold a larger lead of >5%. Comparably, the CNN approach [2] also anticipates the whole sequence in one-shot but does so less effectively. A critical difference from ours is that they optimize category and temporal length with the MSE loss, which is known to produce blurred outcomes [13]. Instead, we directly work in the discrete domain with adversarial learning for sharper outcomes, which provides further support for our discrete representation of time.

If we choose the best results across multiple samples, our accuracy is further boosted by ≈6.5%, which while an improvement over mean accuracy, is not as large as for Breakfast and not as good as the best max results [1]. The difference likely owes to 50Salads being much less multimodal compared to Breakfast and the 50Salads dataset being too small (merely 160 training samples) for learning GANs well. Although our RNN model has similar structure to [1], the discriminator incurs extra parameters and therefore requires more training data.

Epic-Kitchens dataset Figure 3a shows results for Epic-Kitchens as prediction accuracy at specific times in seconds, *i.e*. (1, 3, 5, 7, 10, 20, 30, 40, 50, 60) with both backbones. Our LSTM results are the top performer when prediction time is up to 20 seconds. Afterward, we are on par with TOS [17] at 30s, but then trail by an average of ≈1.7% out to 60s. The largest difference happens at 60s, where the gap is 3%. Resembling observations by comparable work in related areas [55,39,15,8], we find the quality of generated samples decreases as the required prediction length increases. Still, our approach outperforms classic RNN and CNN approaches at all prediction times. It is notable that TOS uses a joint learning of long-term prediction with recognition of current observations [17], whereas RNN/CNN [2] and ours solely focus on sequential modelling. These results might indicate that the auxiliary term helps stablize long-term performance. Our RNN results outpeform alternatives (except our LSTM) only at one temporal position, *i.e*. 1s, and otherwise are analogous to the alternatives [2].

Overall, experiments on all three datasets show that both our models (LSTM and RNN) set new state-of-the-art performance under most protocols. The exception is at distant predictions (*e.g*. 60s in Fig. 3a), implying the necessity of a better temporal model (LSTM) for longer sequences. In the following, we restrict further experiments to the LSTM model due to its generally better performance.

4.4 Analysis of adversarial learning

We now show the benefits of adversarial learning by examining the accuracy curve at various training stages using Epic-Kitchens, reporting averages across 16 samples. We use Epic-Kitchens as our above experiment with this dataset showed the largest performance change with prediction time; Fig. 3b has results. Just after MLE pretraining, accuracy is lower than our comparison approaches, *c.f.* Fig. 3a. As we start adversarial learning, accuracy immediately rises and does so essentially monotonically with increased training epochs for lower prediction times, and begins to converge at 1200 epochs. For epochs beyond 1500, there is evidence of overfitting at longer prediction times, as performance decreases.

4.5 Analysis of number of samples

We now examine the influence of the number of samples on accuracy using the Breakfast dataset as it is larger than the others; see Table 4. We see the largest accuracy increases occur as samples go 2 to 8 and (less so) 8 to 16. Afterwards, improvements grow much slower and performance is stable. In accord with Sec. 4.4 results, adjacent predictions (*e.g.* 10%) benefit more than remote ones (*e.g.* 20-50%). Empirically, we find 16 samples work for both accuracy and efficiency.

Observation	20%				30%			
Prediction	10%	20%	30%	50%	10%	20%	30%	50%
2 Samples	0.6157	0.4525	0.3541	0.2382	0.6408	0.4839	0.3775	0.2570
8 Samples	0.7060	0.5950	0.4395	0.3927	0.7032	0.6564	0.5716	0.4320
16 Samples	0.7222	0.6240	0.5622	0.4595	0.7414	0.7132	0.6530	0.5238
32 Samples	0.7464	0.6319	0.5475	0.4643	0.7611	0.7155	0.6566	0.5219
64 Samples	0.7572	0.6325	0.5547	0.4674	0.7758	0.7150	0.6650	0.5291

Table 4: Influence of various number of samples for Breakfast Dataset, Protocol 1, with dense anticipation mean over classes (MoC) accuracy under Avg metric and LSTM backbone.

4.6 Analysis of normalized distance regularization and diversity

We now provide experimental analysis of the influence our model has on the quality-diversity trade-off. Here, we use the Breakfast dataset because of its large number of videos and relatively high diversity. To evaluate quality, we adopt the accuracy metric, as in Table 2. To evaluate diversity, we calculate the averaged pairwise cosine distance amongst 10 samples, *c.f.* [53] as well as LL, as in Table 2. Both metrics are presented for pure adversarial learning (Adv) and with the aide of normalized distance based adversarial learning (Adv + NormDist); Table 5 has results. With pure adversarial learning (*i.e.* no normalized distance regularizer), too small or too large of a temperature, *e.g.* $\alpha \in \{0.1, 10, 100, 1000\}$ leads to unsatisfactory results. Best results are obtained when α=1; however, the diversity score is not as good as with higher α and the quality score merely equals that of deterministic APP-LSTM in Table 2, row 2.

After joint training with the distance regularizer, quality and diversity scores grow together. Especially, when α=1 the model receives the most benefit in terms of quality and diversity (+2% for LL, +0.138 for cosine distance and +7.89% for accuracy), while with α=10 the model yields smaller improvements. The reason for this pattern is that large α values already yield adequate diversity; so, there is little for the distance regularizer to offer. In the case of $\alpha \geq 10$, however,

Temperature	Loss	LL	Diversity	Accuracy
$\alpha = 0.1$	Adv	-10.83	0.033	23.01
$\alpha = 1$	Adv	**-6.77**	0.124	**60.98**
$\alpha = 10$	Adv	-8.42	0.345	35.19
$\alpha = 100$	Adv	-8.52	0.423	33.78
$\alpha = 1000$	Adv	-11.46	0.582	19.97
$\alpha = 0.1$	Adv + NormDist	-9.04	0.062 (+0.029)	27.61 (+4.60)
$\alpha = 1$	Adv + NormDist	**-4.83**	0.262 (+0.138)	**68.96** (+7.89)
$\alpha = 10$	Adv + NormDist	-8.19	0.358 (+0.013)	36.64 (+1.45)

Table 5: Comparison across various temperatures as well as impact of the normalized distance regularizer. + indicates improvement of adding the normalized distance regularizer (NormDist) to pure adversarial learning (Adv). LSTM backbone is adopted.

LL suffers noticably. Theoretically, small temperature yields accurate one-hot approximation; however, in our experiments α=0.1 does not perform better, possibly because even with our regularizer, gradient variance is too high.

4.7 Visualization of diversity and quality

Figure 4 shows sampled outcomes for our full approach *vs*. our approach lacking normalized distance regularization. Without regularization the small temperature ($\alpha = 1$) samples lack diversity (*i.e*. *Take_butter* is always generated after *Cut_bun*), whereas large temperature ($\alpha = 100$) samples are diverse to the point of being unrealistic (*e.g*. *Add_teabag* generated after *Cut_bun*), *i.e*. they lack quality. In contrast, our full approach gives both plausible and diverse outputs.

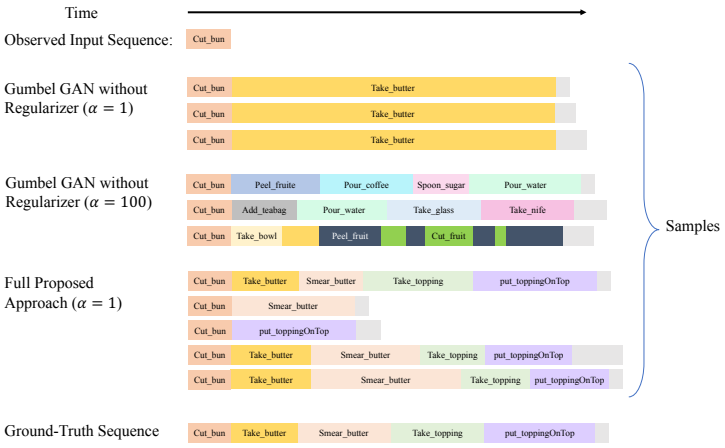


Fig. 4: Visualization of multimodal samples from our full model *vs*. our model lacking normalized distance regularization. See Supplemental Material for additional examples.

5 Summary

In activity anticipation, it is desirable to produce results that realistically capture the potentially multimodal distribution of future actions given initial observations. We have responded with a novel model combining a sequential GAN (for realistic predictions) with a sample distance regularizer (for diverse predictions). By unifying action and time representations, the proposed model can produce outputs that are both realistic and diverse, thereby yielding new state-of-the-art performance on standard activity anticipation datasets. We demonstrated our approach with two different sequence generation backbones (RNN and LSTM).

References

1. Abu Farha, Y., Gall, J.: Uncertainty-aware anticipation of activities. In: ICCVW (2019)
2. Abu Farha, Y., Richard, A., Gall, J.: When will you do what?-Anticipating temporal occurrences of activities. In: CVPR (2018)
3. Babaeizadeh, M., Finn, C., Erhan, D., Campbell, R.H., Levine, S.: Stochastic variational video prediction. arXiv preprint arXiv:1710.11252 (2017)
4. Carreira, J., Zisserman, A.: Quo vadis, action recognition? A new model and the kinetics dataset. In: CVPR (2017)
5. Dai, B., Fidler, S., Urtasun, R., Lin, D.: Towards diverse and natural image descriptions via a conditional gan. In: ICCV (2017)
6. Damen, D., Doughty, H., Maria Farinella, G., Fidler, S., Furnari, A., Kazakos, E., Moltisanti, D., Munro, J., Perrett, T., Price, W., et al.: Scaling egocentric vision: The EPIC-KITCHENS dataset. In: ECCV (2018)
7. Dauphin, Y.N., Fan, A., Auli, M., Grangier, D.: Language modeling with gated convolutional networks. In: ICML (2017)
8. Fedus, W., Goodfellow, I., Dai, A.M.: Maskgan: better text generation via filling in the_. arXiv preprint arXiv:1801.07736 (2018)
9. Feichtenhofer, C., Pinz, A., Wildes, R.: Spatiotemporal residual networks for video action recognition. In: NIPS (2016)
10. Furnari, A., Battiato, S., Maria Farinella, G.: Leveraging uncertainty to rethink loss functions and evaluation measures for egocentric action anticipation. In: ECCVW (2018)
11. Gao, J., Yang, Z., Nevatia, R.: Red: Reinforced encoder-decoder networks for action anticipation. In: BMVC (2017)
12. Glorot, X., Bordes, A., Bengio, Y.: Deep sparse rectifier neural networks. In: AISTAT (2011)
13. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial nets. In: NIPS (2014)
14. Gumbel, E.J.: Statistical theory of extreme values and some practical applications: a series of lectures, vol. 33. US Government Printing Office (1948)
15. Guo, J., Lu, S., Cai, H., Zhang, W., Yu, Y., Wang, J.: Long text generation via adversarial training with leaked information. In: AAAI (2018)
16. Jang, E., Gu, S., Poole, B.: Categorical reparameterization with gumbel-softmax. In: ICLR (2017)
17. Ke, Q., Fritz, M., Schiele, B.: Time-conditioned action anticipation in one shot. In: CVPR (2019)
18. Kim, Y.: Convolutional neural networks for sentence classification. arXiv preprint arXiv:1408.5882 (2014)
19. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
20. Kong, Y., Tao, Z., Fu, Y.: Deep sequential context networks for action prediction. In: CVPR (2017)
21. Kong, Y., Tao, Z., Fu, Y.: Adversarial action prediction networks. IEEE TPAMI (2019)
22. Koppula, H.S., Saxena, A.: Anticipating human activities using object affordances for reactive robotic response. IEEE TPAMI **38**(1), 14–29 (2015)
23. Kuehne, H., Arslan, A., Serre, T.: The language of actions: Recovering the syntax and semantics of goal-directed human activities. In: CVPR (2014)

24. Lai, S., Xu, L., Liu, K., Zhao, J.: Recurrent convolutional neural networks for text classification. In: AAAI (2015)
25. Lea, C., Flynn, M.D., Vidal, R., Reiter, A., Hager, G.D.: Temporal convolutional networks for action segmentation and detection. In: CVPR (2017)
26. Ledig, C., Theis, L., Huszár, F., Caballero, J., Cunningham, A., Acosta, A., Aitken, A., Tejani, A., Totz, J., Wang, Z., et al.: Photo-realistic single image super-resolution using a generative adversarial network. In: CVPR (2017)
27. Lee, A.X., Zhang, R., Ebert, F., Abbeel, P., Finn, C., Levine, S.: Stochastic adversarial video prediction. arXiv preprint arXiv:1804.01523 (2018)
28. Li, Y., Du, N., Bengio, S.: Time-dependent representation for neural event sequence prediction. ICLRW (2018)
29. Liang, J., Jiang, L., Murphy, K., Yu, T., Hauptmann, A.: The garden of forking paths: Towards multi-future trajectory prediction. In: CVPR. pp. 10508–10518 (2020)
30. Liang, J., Jiang, L., Niebles, J.C., Hauptmann, A.G., Fei-Fei, L.: Peeking into the future: Predicting future person activities and locations in videos. In: CVPRW. pp. 5725–5734 (2019)
31. Liu, S., Zhang, X., Wangni, J., Shi, J.: Normalized diversification. In: CVPR (2019)
32. Lukas, N., Andrew, Z., Vedaldi, A.: Future event prediction: If and when. In: CVPRW. pp. 14424–14432 (2019)
33. Maddison, C.J., Mnih, A., Teh, Y.W.: The concrete distribution: A continuous relaxation of discrete random variables. In: ICLR (2017)
34. Mahmud, T., Billah, M., Hasan, M., Roy-Chowdhury, A.K.: Captioning near-future activity sequences. arXiv preprint arXiv:1908.00943 (2019)
35. Mahmud, T., Hasan, M., Roy-Chowdhury, A.K.: Joint prediction of activity labels and starting times in untrimmed videos. In: ICCV (2017)
36. Mathieu, M., Couprie, C., LeCun, Y.: Deep multi-scale video prediction beyond mean square error. arXiv preprint arXiv:1511.05440 (2015)
37. Mehrasa, N., Jyothi, A.A., Durand, T., He, J., Sigal, L., Mori, G.: A variational auto-encoder model for stochastic point processes. In: CVPR (2019)
38. Mohamed, A., Qian, K., Elhoseiny, M., Claudel, C.: Social-stgcnn: A social spatio-temporal graph convolutional neural network for human trajectory prediction. In: CVPR. pp. 14424–14432 (2020)
39. Nie, W., Narodytska, N., Patel, A.: Relgan: Relational generative adversarial networks for text generation. In: ICLR (2019)
40. Oord, A.v.d., Dieleman, S., Zen, H., Simonyan, K., Vinyals, O., Graves, A., Kalchbrenner, N., Senior, A., Kavukcuoglu, K.: Wavenet: A generative model for raw audio. arXiv preprint arXiv:1609.03499 (2016)
41. Santoro, A., Faulkner, R., Raposo, D., Rae, J., Chrzanowski, M., Weber, T., Wierstra, D., Vinyals, O., Pascanu, R., Lillicrap, T.: Relational recurrent neural networks. In: NIPS (2018)
42. Schydlo, P., Rakovic, M., Jamone, L., Santos-Victor, J.: Anticipation in human-robot cooperation: A recurrent neural network approach for multiple action sequences prediction. In: ICRA (2018)
43. Shi, Y., Fernando, B., Hartley, R.: Action anticipation with RBF kernelized feature mapping rnn. In: ECCV (2018)
44. Simonyan, K., Zisserman, A.: Two-stream convolutional networks for action recognition in videos. In: NIPS (2014)
45. Singh, G., Saha, S., Cuzzolin, F.: Predicting action tubes. In: ECCVW (2018)
46. Singh, G., Saha, S., Sapienza, M., Torr, P.H., Cuzzolin, F.: Online real-time multiple spatiotemporal action localisation and prediction. In: ICCV (2017)

47. Stein, S., McKenna, S.J.: Combining embedded accelerometers with computer vision for recognizing food preparation activities. In: UbiComp (2013)
48. Sun, C., Shrivastava, A., Vondrick, C., Sukthankar, R., Murphy, K., Schmid, C.: Relational action forecasting. In: CVPR. pp. 273–283 (2019)
49. Sutskever, I., Vinyals, O., Le, Q.V.: Sequence to sequence learning with neural networks. In: NIPS (2014)
50. Veselỳ, K., Ghoshal, A., Burget, L., Povey, D.: Sequence-discriminative training of deep neural networks. In: Interspeech (2013)
51. Vondrick, C., Pirsiavash, H., Torralba, A.: Anticipating visual representations from unlabeled video. In: CVPR (2016)
52. Wang, L., Xiong, Y., Wang, Z., Qiao, Y., Lin, D., Tang, X., Van Gool, L.: Temporal segment networks: Towards good practices for deep action recognition. In: ECCV (2016)
53. Yang, D., Hong, S., Jang, Y., Zhao, T., Lee, H.: Diversity-sensitive conditional generative adversarial networks. In: ICLR (2019)
54. Yeh, R.A., Schwing, A.G., Huang, J., Murphy, K.: Diverse generation for multi-agent sports games. In: CVPR. pp. 4610–4619 (2019)
55. Yu, L., Zhang, W., Wang, J., Yu, Y.: Seqgan: Sequence generative adversarial nets with policy gradient. In: AAAI (2017)
56. Zhao, H., Wildes, R.P.: Spatiotemporal feature residual propagation for action prediction. In: ICCV (2019)
57. Zhu, J.Y., Park, T., Isola, P., Efros, A.A.: Unpaired image-to-image translation using cycle-consistent adversarial networks. In: ICCV (2017)