BTGym formalism [draft]

Andrew Muzikin, muzikinae@gmail.com

January 7, 2018

Abstract

Conceptual and formal definitions behind BTGym package, very incomplete and under developmnet. Please do not cite or distribute.

1 Data

Our data space is discrete timeflow of equitype records. For convenience of defining episodic MDP over such data and setting problem objective it should be somehow structured. Here data related concepts and notation are introduced: from single data record to episode, trial and dataset.

Let single timerecord d be a tuple of discrete datetime stamp and value vector of arbitrary length:

$$d \doteq \langle t, v \mid t \in DiscreteTime, v \in \mathbf{R}^m \rangle$$

Define episode as tuple of time-ordered set of timerecords starting at time $t = a$, finishing at $t = b$ of maximum number of records M and simple sampling function:

$$e^{a,b,M} \doteq \langle d^{a,b,M}, s(t) \rangle$$

$$d^{a,b,M} \doteq \{d_i \mid \forall i = 1, .., M : a \leq t[d_{i-1}] < t[d_i] \leq b\}$$

$$s(t) \doteq \begin{cases} d(t), t \in [a, b] \\ terminal, otherwise \end{cases}$$

Minimum and maximum timestamps for episode:

$$t_{min} = a, t_{max} = b$$

For any two episodes define a relation of precedence:

$$e \prec h \Leftrightarrow t^e_{max} < t^h_{min}$$

A 'time a to time b' set consisiting of N episodes:

$$E^{a,b,N} = \{e_i \mid \forall i = 1, .., N : a \leq t^{e_i}_{min} \leq t^{e_i}_{max} \leq b\}$$

For any such set we can define upper and lower elements:

$$\forall E \exists e \in E : e \doteq e_{inf} \Leftrightarrow \forall h \in E : t^e_{min} \leq t^h_{min}$$

$$\forall E \exists e \in E : e \doteq e_{sup} \Leftrightarrow \forall h \in E : t^e_{max} \geq t^h_{max}$$

Realtion of precedence for two sets of episodes:

$$E \prec H \Leftrightarrow e_{sup} \prec h_{inf} \mid e_{sup} \in E, h_{inf} \in H$$

Let $Trial$ object be a tuple of disjoint or preceding train and test sets and sampling probability functions, inducing distributions over time intervals:

$$T \doteq \langle E^{Train} \cup E^{Test}, q^{Train}(e), q^{Test}(e) \mid E^{Train} \prec E^{Test} or E^{Train} \cap E^{Test} = \emptyset \rangle$$

$$q^{Train}(e) \doteq \begin{cases} Beta_t(\alpha, \beta), e \in E^{Train} \\ 0, otherwise \end{cases} \qquad q^{Test}(e) \doteq \begin{cases} Uniform_t, e \in E^{Test} \\ 0, otherwise \end{cases}$$

For brewity we refer to train and test subsets of a trial as T_{Train}, T_{Test}. Making train episodes beta-distributed over timescale is pure heuristic that allows skewed sampling, e.g. for prioritising recent episodes. Test episodes are always sampled uniformly.

Finally let $DataDomain$ object be a tuple of set of L trials in $[a, b]$ time interval and heuristic sampling probability function:

$$D \doteq \langle D^{a,b,L}, p(T) \rangle,$$

$$D^{a,b,L} \doteq \{T_i \mid \forall i = 1, .., L \forall e_{inf}, e_{sup} \in E^{Train}_i \cup E^{Test}_i : a \leq t^{e_{inf}}_{min} < t^{e_{sup}}_{max} \leq b\}$$

for random case:

$$p(T) = Uniform_t, T \in D^{a,b,L}$$

for sequential case:

$$p(T, i) = \begin{cases} 1, T = T_i \in D^{a,b,L} \\ 0, otherwise \end{cases}$$

One can refer to train data as known data and test data as unknown, future data. While we require train interval strictly precede test one, for train data itself there is no such restriction.

We refer to *source* data domain as one consisting of fully known trials (test and train data sre given) an *target* data domain where only train part of the trial is available for training.

Review from top to bottom:

- Sampling (randomly or sequentiall) from *datadomain* object returns a *datatria*;
- Given a *trial* one can sample either train or test *episode*. Test *episodes* are sampled unifromly from test interval while train samples can be beta-distributed on train interval;
- *Records* in the *episode* are ordered and retrieved by *timeindex*.

2 MDP

An MDP is defined by a tuple $\langle S, A, R, P, \gamma \rangle$, which consists of a set of states S, a set of actions A, a reward function $R(s, a)$, a transition function $P(s, a, s') = \rho(s' \mid s, a)$, and a discount factor γ. In each state $s \in S$, the agent takes an action $a \in A$. Upon taking this action, the agent receives a reward $R(s, a)$ and reaches a new state s', determined from the probability distribution $P(s' \mid s, a)$. A policy π maps mapping states to actions the agent will take. We consider episodic environments with finite horisont, discrete action sapce. Each episode is taking place in discrete time interval $[a, b]$ of length N. The goal of the agent is to find an optimal policy π^* that maximizes the expected discounted total reward over the agents lifetime. In practice policy is usually restricted to class of differentiable parametrized functions, $\pi \doteq \pi(\Theta)$.

Specifically for BTgym, it is sensible to define *state* as tuple of at least two sub-states, $s = \langle s_{ext}, s_{int} \rangle$. External state s_{ext} encloses information of part of environment agent actions are not supposed to affect, e.g. market prices, news feeds etc. Internal state s_{int}, in the opposite, presents part of environment affected by agents actions: broker account value, postition size, etc. BTGym problem formulation.

BTgym problem domain can be expressed via set of MDPs induced by external data doamin $D^{a,b,L}$. Indeed, restricting to same A, R, γ, let all MDPs variate by S, P and later only depend on train/test data provided; in fact it is convinient to define state space and transition function such as all MDPs will duffer only in trasnsition probability functions. Than results from [1] can be applied. Let state space be a union of all int/ext tuples induced by data:

$$S^D \doteq \{\langle s_{ext}, s_{int} \rangle \mid s_{ext} \in e \in T^{Train} \cup T^{Test}, T \in D^{a,b,L}\}$$

Each trial from D induce two subspaces in S^D:

$$\forall T \in D^{a,b,L}, \forall e \in T :$$

$$S^{Train} \doteq \{\langle s_{ext}, s_{int} \rangle \mid s_{ext} \in e \in T^{Train}\}$$

$$S^{Test} \doteq \{\langle s_{ext}, s_{int} \rangle \mid s_{ext} \in e \in T^{Test}\}$$

Let corresponding transition functions be zero everywhere in S except for these subspaces:

$$P^{Train} \doteq \begin{cases} \rho(s' \mid s, a) \mid s, s' \in S^{Train} \\ 0, otherwise \end{cases}$$

$$P^{Test} \doteq \begin{cases} \sigma(s' \mid s, a) \mid s, s' \in S^{Test} \\ 0, otherwize \end{cases}$$

Than two task-induced MDPs will became:

$$m_{Train}(T) \doteq m(T^{Train}) \doteq \langle S, A, R, P^{Train}, \gamma \rangle$$

$$m_{Test}(T) \doteq m(T^{Test}) \doteq \langle S, A, R, P^{Test}, \gamma \rangle$$

Let *task* be an tuple of MDPs induced by single data *trial* and define *domain* to be set of all such tasks over data:

$$m(T) \doteq \langle m_{Train}, m_{Test} \rangle (T)$$

$$M^D \doteq M(D) \doteq \{m(T) \mid T \in D\}$$

3 Objective

The final goal is, given *source* task domain induced by fully known *source* data domain $M^D_{Source} \leftarrow D_{Source}$, a *target* task domain $M^D_{Target} \leftarrow D_{Target}$, induced by target data domain (i.e. one where test data not known in advance), maximize expected total reward over test tasks in target domain: $m_{Test} \in M^D_{Target}$.

Note that transition probability function of interest P^{Test}_{Target} is unknown and defined 'on the fly' as new data arrive while $P^{Train}_{Source}, P^{Test}_{Source}$ and P^{Train}_{Target} are known.

Thus, target optimal policies for test tasks $\mu^*(m_{Test}), m \in M^D_{Target}$ can only be inferred from target train policies $\mu(m_{Train}), m \in M^D_{Target}$ and source ones $\pi(m_{Train}), \pi(m_{Test}), m \in M^D_{Source}$.

References

[1] Balazs Csanad Csaji et al., "Value Function Based Reinforcement Learning in Changing Markovian Environments," in *Journal of Machine Learning Research 9 (2008)*