2020 16th IEEE International Conference on Automation Science and Engineering (CASE)
August 20-21, 2020, Online Zoom Meeting

Risk-aware Energy Management of Extended Range Electric Delivery Vehicles with Implicit Quantile Network

Pengyue Wang, Yan Li, Shashi Shekhar, *Fellow, IEEE,* William F. Northrop*

***Abstract*—Model-free reinforcement learning (RL) algorithms are used to solve sequential decision-making problems under uncertainty. They are data-driven methods and do not require an explicit model of the studied system or environment. Because of this characteristic, they are widely utilized in Intelligent Transportation Systems (ITS), as real-world transportation systems are highly complex and extremely difficult to model. However, in most literature, decisions are made according to the expected long-term return estimated by the RL algorithm, ignoring the underlying risk. In this work, a distributional RL algorithm called implicit quantile network is adapted for the energy management problem of a delivery vehicle. Instead of only estimating the expected long-term return, the full return distribution is estimated implicitly. This is highly beneficial for applications in ITS, as uncertainty and randomness are intrinsic characteristics of transportation systems. In addition, risk-aware strategies are integrated into the algorithm with the risk measure of conditional value at risk. In this study, we demonstrate that by changing a hyperparameter, the trade-off between fuel efficiency and the risk of running out of battery power during a delivery trip can be controlled according to different application scenarios and personal preferences.**

I. Introduction

Many transportation-related applications involve processes of sequential decision-making, such as autonomous driving [1][2], traffic light cycle control [3][4] or energy management (EMS) of hybrid vehicles [5][6]. Optimal control methods such as dynamic programming are perfect solutions if the mathematical model of the studied system is given or the environment can be accurately modeled [7]. However, a real-world transportation system is highly complex to model and has a stochastic nature due to some physical systems and human behaviors. So, with the coming era of big data, data-driven, model-free reinforcement learning (RL) algorithms [8] promise to become more and more popular in the area of Intelligent Transportation Systems (ITS) research, with emerging technologies like Vehicle-to-Vehicle (V2V), Vehicle-to-Cloud (V2C) and Vehicle-to-Infrastructure (V2I) communications.

The core idea of standard RL algorithms is the expected long-term return [8]. For value-based methods, decisions at each timestep are made according to the estimated expected long-term return for each action. For policy-based methods, a gradient ascent step is performed on the parametrized policy according to the expected value. Although researchers have achieved some notable success using standard RL algorithms in video gaming and robotics control, these algorithms are not suitable for safety-critical applications. This is because expected value loses important information about the future return, such as the spread [9][10]. For example, if the return distribution is multi-modal or has a high variance, only considering the expected value may lead to bad decisions in a real-world scenario. This is because what the agent will receive during one task is just one sample from the distribution. In Fig. 1, it can be observed that although the expected values of action 2 in both cases are higher than action 1, action 2 may lead to much worse results.

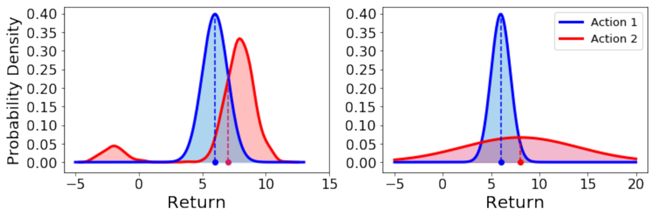

Fig. 1. Multi-modality (left) and high variance (right) return distributions.

This work adapts a distributional RL algorithm called implicit quantile network (IQN) [11] into an energy management problem of extended range electric vehicles (EREVs) used for package delivery. A risk-aware strategy is integrated into this algorithm based on an implicitly estimated return distribution and a risk measure called conditional value at risk (CVaR) [12]. We show that the trade-off between fuel economy and the risk of running out of battery during a delivery task can be controlled by a hyperparameter in CVaR. The proposed framework is illustrated in Fig. 2.

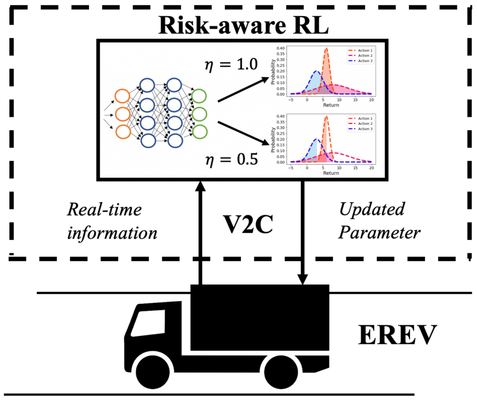

Fig. 2. Illustration of the proposed framework. η is the hyperparameter related to CVaR with a range of (0,1].

*Corresponding author

The information, data, or work presented herein was funded in part by the Advanced Research Projects Agency-Energy (ARPA-E) U.S. Department of Energy, under Award Number DE-AR0000795.

P. Wang, Y. Li, S. Shekhar and W. F. Northrop are with University of Minnesota, Minneapolis, MN 55455. (email: wang6609@umn.edu; lixx4266@umn.edu; shekhar@umn.edu; wnorthro@umn.edu)

II. Related Work

There are two kinds of uncertainty in an RL framework: model uncertainty and risk. [13][14]. Model uncertainty (also called epistemic uncertainty) can be explained away with enough data, and it is usually handled under a Bayesian framework [15]. It can be beneficial to RL applications in at least two aspects. First, it helps with exploration during the training process. In [16], a Bayesian ensemble of neural networks (NNs) was used to quantify the predictive uncertainty and facilitate the exploration strategy of Thompson sampling. Second, model uncertainty can help identify novel/out-of-distribution data during testing, providing a safer algorithm. In [1], an autonomous driving system was able to identify novel obstacles during testing and perform safer and more conservative actions. Although model uncertainty is an important topic, it should be distinguished clearly from risk, which is the topic of this work.

Risk, also called aleatoric uncertainty [13], is an intrinsic property of a system. For example, no matter how many times we flip a coin, the results are still stochastic. In a standard RL algorithm, the decisions are made based on the expected long-term return, which means if we have many chances to choose an action at a certain state, it will lead to the highest possible return on average. However, for real-world transportation applications, it is not enough to deliver an algorithm that works well on average, especially for safety-critical applications. Therefore, it is very important to be able to estimate the full return distribution, not just one expected return value. C51 [19], the first distributional RL algorithm integrated into the popular DQN algorithm [17] achieved state-of-the-art performance on video games. It calculates the probabilities for a set of predefined returns. Another approach used quantile regression (QR) to calculate a fixed number of quantiles of a return distribution with fixed probabilities [10]. This method achieved better theoretical guarantee in convergence compared with C51. Implicit quantile network (IQN) [11], the method used in this paper, removes the limitation of a fixed number of quantiles.

Energy management problems of hybrid vehicles has been a popular research area for a long time. A detailed survey can be found in [18]. RL-based algorithms have also been applied in this area with different degrees of success [19][20]. However, there is little work that considers the issue of risk in these methods. One of the main reasons risk has been ignored is that for most hybrid electric vehicles, the worst-case result is simply bad energy efficiency, which does not lead to more serious problems. However, for our EREVs, a bad decision can lead to the condition of running out of battery power, which will delay the delivery tasks and can be very dangerous, for example, if it happens on a highway. In our previous work [21], the high variance and multi-modal return distributions were shown using the algorithm of C51. This means decisions only based on expected values may lead to much worse results on individual tasks. In this work, we build a risk-aware strategy based on IQN that selects actions according to a risk measure called CVaR which includes the expected value as a special case.

III. Reinforcement Learning Problem Formulation

A. Vehicle Configuration and EMS Problem

The powertrain configuration of the studied EREV is shown in Fig. 3. The driving force is provided by an electric motor which uses the energy from a high-capacity battery (56 kWh). The internal combustion engine serves as a range-extender used to charge the battery according to the EMS. There is no mechanical connection between the engine output shaft and the wheels so that they are completely decoupled.

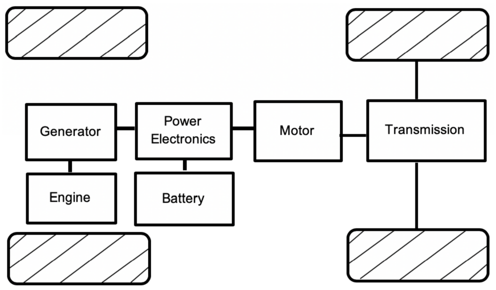

Fig. 3. Powertrain configuration of the EREV.

The in-use EMS is a rule-based method. Under this strategy, the engine only operates at one predefined high-efficiency speed and load condition to provide reliable, safe and low-noise operation (11kW). To achieve high fuel efficiencies, ideally, no fuel should be used for short trips that do not exceed the all-electric range (AER) of the EREV. For longer trips, the goal is to reach a target end state of charge (SOC) at the end of the trip with the help of the range-extender. The engine control logic is based on the measured real-time SOC during the trip: if it falls lower than a calculated SOC reference value, the engine should turn on. The reference value is calculated as:

$$SOC_{ref} = 100\% \times \left(1 - 0.9\frac{d_t}{L_{set}}\right), \tag{1}$$

where d_t is the distance the EREV has traveled on a given trip, and L_{set} is the energy-compensated expected trip distance. The value of 0.9 comes from the setting that the target end SOC in this work is 10%. In addition, if the calculated SOC_{ref} is higher than 60%, it is set to 60% to prevent fuel use on short trips.

The SOC_{ref} represents how much energy is expected to be left when the EREV has traveled for d_t given a value of L_{set}. The L_{set} is the parameter that needs to be optimized, i.e., the value of L_{set} should change so that the fuel use is minimized and the value of SOC is always higher than 10% during the delivery trip. The main challenge of this problem is that even for one vehicle running in a certain delivery area, the trip distance and energy intensity for different parts of the trip vary day-to-day due to factors like delivery demand, weather and traffic conditions. The distribution of distance and energy intensity for the EREV in this work is shown in Fig. 4. In current practice, to prevent a vehicle from running out of battery with high confidence, the value of L_{set} is always set to high in the fleet's vehicles, which leads to excessive fuel use for most trips.

In Fig. 5., a comparison of SOC and fuel use under two L_{set} settings is shown. It can be observed that for the lower

L_{set} setting, less fuel is used and the SOC reaches the value of 10% at the end of trip. However, this outcome cannot be achieved for future trips as no detailed information is available about the conditions that made this outcome possible. Consequently, an RL algorithm is used to update the value of L_{set} during the delivery trip, using the real-time information and a strategy learned from historical trips.

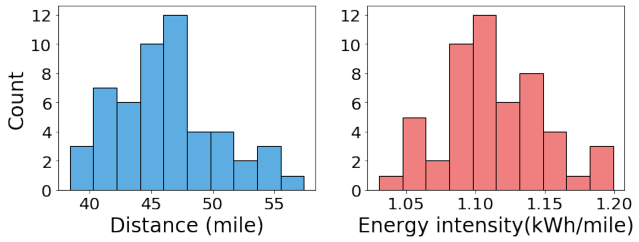

Fig. 4. Distribution of distance and energy intensity for the used EREV.

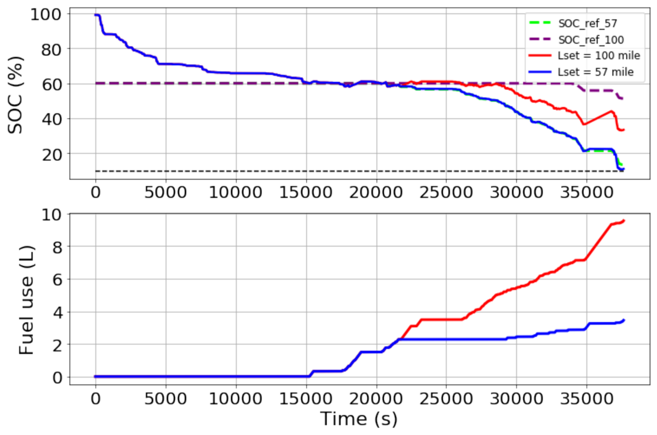

Fig. 5. Comparison of SOC_{ref} and real-time SOC (upper) and fuel use (lower) with different L_{set} settings for one measured delivery trip.

B. Formulating the EMS Problem as a RL Problem

The goal of RL is to maximize a cumulated reward that an agent receives during its interaction with an environment through a sequential decision-making process enabled by a policy. The RL problem is solved if a good policy is obtained.

During one timestep of interaction, the agent receives the current state information s_t from the environment and executes an action a_t according to the policy $\pi: s_t \to a_t$. The environment then responds with the next state s_{t+1} and a reward r_t. This process continues until a terminal state s_T is achieved:

$$s_0, a_0, r_0, s_1, a_1, r_1, s_2, a_2, r_2 \dots s_T. \quad (2)$$

The cumulated reward at each timestep t is defined as:

$$G_t = r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \cdots + \gamma^{T-1-t} r_{T-1}, \quad (3)$$

where γ is the discount factor.

To formulate our EMS problem as an RL problem, we need to specify a state space, an action space, a transition probability $p(s_{t+1}|s_t, a_t)$, a reward function $r(r_t|s_{t+1}, s_t, a_t)$ and a discount factor γ. γ is set to 0.99 in this work.

1) State space

The state space is a 7-dimensional space consisting of the available real-time information from the EREV during the delivery trip, including traveled time, distance, current SOC, fuel use, GPS position and current L_{set} setting:

$$s_t = [t_{travel}, d, SOC, f, x, y, L_{set}]. \quad (4)$$

2) Action space

The action space is a set of predefined actions:

$$a_t \in [-15, -10, -5, 0, +5, +10, +15], \quad (5)$$

where the values represent the L_{set} change.

3) Transition probability

Transition probability $p(s_{t+1}|s_t, a_t)$ refers to the unknown underlying system that generates the velocity profiles for each delivery trip. It is approximated using a simplified vehicle model and 52 historical delivery trips of the vehicle.

The simplified vehicle model is used to simulate the SOC and fuel consumption for a recorded trip under different L_{set} settings. Given an action at time a_t and current condition s_t, the model can calculate the next state s_{t+1}. A detailed introduction to the model can be found in [5].

For each completed delivery trip, the required parameters such as velocity, GPS coordinates and SOC were recorded with timestamps when the vehicle was running. The distance and energy intensity of the trips ranged from 38 to 57 miles and 1.03 to 1.20 kWh/miles, respectively.

4) Reward function

The reward function is designed as:

$$r_t = r_f t_{f,t} + r_{SOC} t_{SOC,t} + r_{a,t} + r_c. \quad (6)$$

The first term penalizes fuel use, and its magnitude is proportional to the engine running time with a coefficient of -0.001. The second term penalizes the condition of SOC lower than 10%, and its magnitude is proportional to the time in that condition with a coefficient of -0.060. The third term encourages the algorithm to find a policy that changes the L_{set} as less frequently as possible. It can be expressed as $r_{a,t} = -0.020\mathbb{I}_{a_t \neq 0}$, where $\mathbb{I}_{a_t \neq 0}$ equals 1 if $a_t \neq 0$. The last term is used to compensate for the negative reward caused by the necessary fuel use. It is only given at the end of the trip, since it is only at that time that the minimum amount of fuel to keep the SOC always higher than 10% can be calculated.

IV. Risk-aware RL Algorithm

A. Implicit Quantile Network

The key to a standard value-based RL algorithm is the action-value function [8]:

$$Q_\pi(s, a) = E_\pi[G_t | s_t = s, a_t = a], \quad (7)$$

which represents the expected long-term return G_t that can be achieved if action a is taken at state s and following policy π thereafter. The optimal action-value function for a given problem is:

$$Q^*(s, a) = max_\pi Q_\pi(s, a), \quad (8)$$

which represents the highest action-value that can be achieved for all possible state-action pairs under all possible policies.

The optimal control policy can be derived easily from the optimal action-value function by acting greedily with respect to it:

$$\pi^*(s) = argmax_a Q^*(s,a). \qquad (9)$$

The main purpose of a standard value-based deep RL algorithms is to estimate the $Q^*(s,a)$ with a NN parametrized by ψ, which is also called a Q-network, $Q(s,a;\psi)$.

In this work, we adapt a distributional RL algorithm called implicit quantile network to the EMS problem and model the full return distribution of given state-action pairs implicitly. The IQN parametrized by θ takes a state-action pair (s,a) and a sample τ from a uniform distribution $U([0,1])$ as inputs and outputs a sample $Z_\tau(s,a;\theta)$from the implicitly defined return distribution $Z(s,a;\theta)$. The action-value $Q(s,a;\theta)$ can be estimated by multiple samples of $Z_\tau(s,a;\theta) \sim Z(s,a;\theta)$ as:

$$Q(s,a;\theta) = E_{\tau\sim U([0,1])}[Z_\tau(s,a;\theta)] \approx \frac{1}{K}\sum_{i=1}^{K} Z_{\tau_i}(s,a;\theta), (10)$$

where index i represents the ith sample from $Z(s,a;\theta)$, and K represents the total number of samples. Fig. 6 illustrates the difference between a standard DRL algorithm, DQN [17], and the IQN.

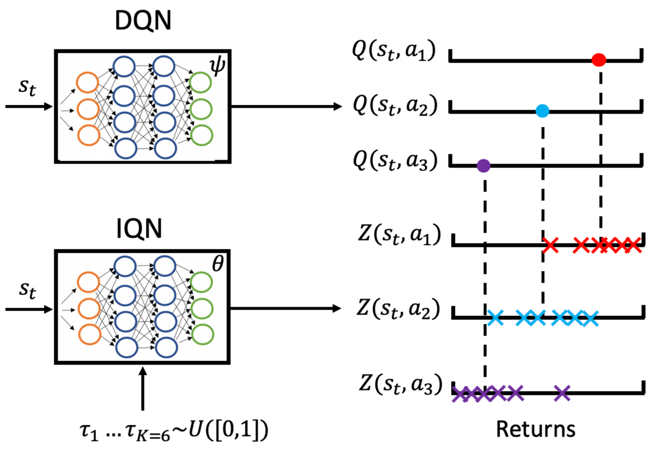

Fig. 6. Illustration of the differences between DQN and IQN. The DQN outputs the expected value for a given state-action pair while IQN outputs samples from the implicitly modeled return distribution whose mean is the output of DQN.

The return distribution is modeled implicitly by estimating the quantile function $QF_{Z(s,a)}(\tau)$, also known as the inverse cumulative distribution function (inverse CDF). Assume the CDF of the return distribution is $F_{Z(s,a)}: Z_\tau(s,a) \to [0,1]$, then:

$$Z_\tau(s,a) = QF_{Z(s,a)}(\tau) = F^{-1}_{Z(s,a)}(\tau), \qquad (11)$$

which indicates a mapping from τ sampled from $[0,1]$ to $Z_\tau(s,a)$. A simple example is shown in Fig. 7. Assume the $QF_{Z(s,a)}$ of a return distribution $Z(s,a)$ for a state-action pair is given, sampling τ from uniform distribution $U([0,1])$ and then feed them into $QF_{Z(s,a)}(\tau)$ equals to sampling returns from the underlying return distribution according to its probability density function (blue dashed line). The sampled returns (red dots) can be used to calculate the action-value according to (10).

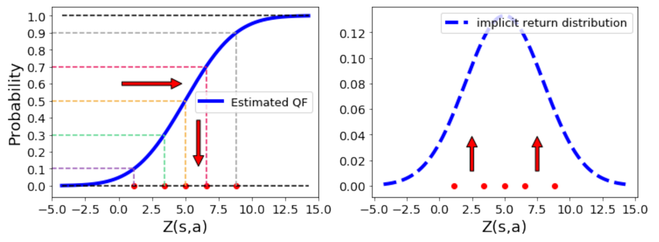

Fig. 7. A simple example illustrating how to sample returns $Z_\tau(s,a)$ from the implicitly modeled return distribution. Assume five τ values: 0.1, 0.3…0.9 are sampled from the uniform distribution $U([0,1])$, five values of $Z_\tau(s,a)$ are calculated by feeding the five τ values into the estimated $QF_{Z(s,a)}(\tau)$. With enough number of samples, the mean and other statistics of the underlying return distribution can be estimated accurately.

The quantile function $QF_{Z(s,a)}(\tau)$ is estimated by IQN. It is trained by the quantile regression (QR) loss function. In Fig. 8, a simple linear example illustrates the difference between common mean squared loss (MSE) and QR loss.

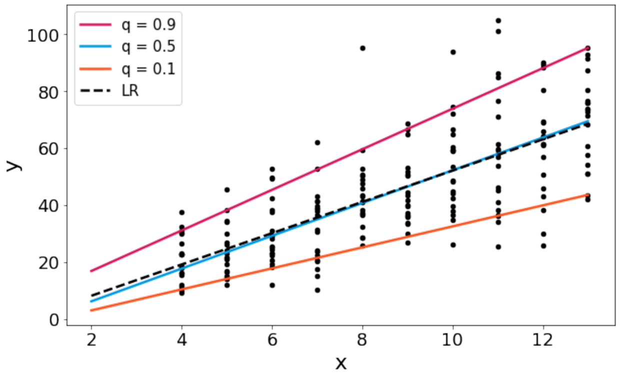

Fig. 8. Comparison of fitting an artificial dataset with linear regression and quantile regression with 0.9, 0.5 and 0.1 quantiles. For each x, twenty y values are sampled from a gamma distribution. For linear regression (LR), the MSE is minimized, which tries to estimate the conditional mean of the underlying assumed Gaussian distribution for each input x. For quantile regression, there is no assumption about the underlying distribution and the conditional quantile values are estimated.

For standard RL algorithms, the square of the temporal difference (TD) error is minimized where the TD error is:

$$\delta_t = r_t + \gamma max_a Q(s_{t+1},a;\psi') - Q(s_t,a_t;\psi). \qquad (12)$$

For IQN, the QR loss of the TD error shown below is minimized:

$$\delta_t^\tau = r_t + \gamma max_a Z_{\tau'}(s_{t+1},a;\theta') - Z_\tau(s_t,a_t;\theta). \qquad (13)$$

The detailed QR loss and MSE for the supervised regression problem and DRL problem is summarized in Table I. The detailed algorithm to train the IQN in the EMS problem is shown in Algorithm 1.

TABLE I
COMPARISON OF QR LOSS AND MSE UNDER TWO PROBLEM SETTINGS

	QR loss	MSE
Regression	$(\tau - \mathbb{1}_{y<wx+b})[y-(wx+b)]$	$[y-(wx+b)]^2$
Deep RL	$(\tau - \mathbb{1}_{\delta_t^\tau<0})\delta_t^\tau$	δ_t^2
Goal	Estimate the τth quantile	Estimate the mean

Algorithm 1 Implicit Quantile Network

```
Initialize IQN and target IQN with same set of random parameters $\theta, \theta'$
Initialize the replay buffer $B$ to capacity $D$ with no transitions
Initialize $\varepsilon = 1.0$, $K = 32$, $D = 5e4$, $n = 48$, $M = 800$, $N = 52$
For epoch = 1, $M$ do
.   For trip = 1, $N$ do
.   .   Get initial state $s_0$ from current trip data
.   .   $t = 0$
.   .   While $t < T$, do
.   .   .   With probability $\varepsilon$, randomly select an action $a_t$
.   .   .   Otherwise select $a_t = argmax_a \sum_{i=1}^{K} Z_{\tau_i}(s_t, a; \theta)$
.   .   .   Update current $L_{set}$ according to $a_t$
.   .   .   Run the vehicle model for 2000s with updated $L_{set}$
.   .   .   Return state $s_{t+1}$ and $r_t$
.   .   .   Store the transition $(s_t, a_t, r_t, s_{t+1})$ in the replay buffer $B$
.   .   .   Sample $n$ transition pairs from $B$
.   .   .   For each transition pair $(s_j, a_j, r_j, s_{j+1})$, do
.   .   .       $Z_{target,j} = \begin{cases} r_j & \text{if trip terminates at step } j+1 \\ r_j + \gamma max_a Z_{\tau_j'}(s_{j+1}, a; \theta) & \text{otherwise} \end{cases}$
.   .   .       $\delta_j^{\tau_j} = Z_{target,j} - Z_{\tau_j}(s_j, a_j; \theta)$
.   .   .   End For
.   .   .   Perform a gradient descent step on $\sum_{j=1}^{n}(\tau_j - \mathbb{1}_{\delta_j^{\tau_j}<0})\delta_j^{\tau_j}$
.   .   .   Perform soft update on the target network
.   .   .   $s_t = s_{t+1}$
.   .   $\varepsilon = \max(0.01, 0.995 \times \varepsilon)$
.   End For
End For
```

B. *Risk Measure*

Conditional value at risk [12] is used as the risk measure in this work. It is defined as:

$$CVaR_\eta(s,a) = E_{r\sim Z(s,a)}\left[r \middle| r < F_{Z(s,a)}^{-1}(\eta)\right], \tag{14}$$

where η is a hyperparameter. If $\eta = 1.0$, the calculated $CVaR_\eta(s,a)$ is the expected value of the return distribution, which is equal to the action value $Q(s,a)$ and leads to a standard risk-neutral strategy. If $\eta < 1.0$, it calculates the expected value for the worst η cases, which leads to risk-averse strategies. A simple example is shown in Fig. 9.

After the IQN is trained, risk-aware strategies can simply sample τ_i from $U([0,\eta])$ and choose actions by:

$$a_t = argmax_a \frac{1}{K}\sum_{i=1}^{K} Z_{\tau_i}(s,a;\theta). \tag{15}$$

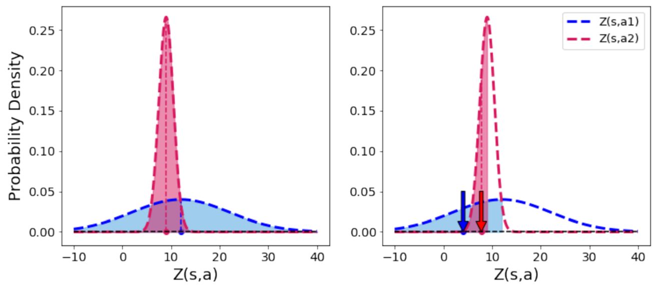

Fig. 9. Illustration of CVaR with $\eta = 1.0$ (left) and $\eta = 0.5$ (right). It can be observed that although the expected value of return distribution $Z(s,a_1)$ is higher than that of $Z(s,a_2)$ (the position of the vertical dashed line), it does not hold true for $\eta = 0.5$ due to its high variance (annotated by arrows).

V. Simulation Results

A feedforward NN was used to build the IQN, following the structure of the original paper [11]. There were two hidden layers in the main structure with 64 and 32 units. The embedding layer for the sampled τ had 64 units. The embedding was combined with the features from the first hidden layer in the main structure by element-wise multiplication. The activation function for all units was ReLU [22]. The optimizer used was Adam [23] with an initial learning rate of 0.0005.

The trained RL algorithm was tested on 58 unseen test trips with a distance range of 35 to 57 miles. For $\eta = 1.0$ (standard risk-neutral strategy), it achieved an average MPGe (mile per gallon gasoline equivalent) of 28.4. This was only 7.1% less efficient than the ideal value of 30.6, which can only be achieved with the full velocity profile information. Our previous work [21] showed similar results. However, this time our focus is the potential benefits of utilizing a risk-aware strategy.

Fig. 10 shows how the trained algorithm performed on a short trip. The L_{set} change at each timestep is very similar under the conditions of $\eta = 1.0$, $\eta = 0.5$ and $\eta = 0.1$, which leads to identical fuel use and SOC curves. This can be explained by the return distribution shown in Fig. 11, which corresponds to the state A annotated in Fig. 10. For all possible actions, the return distributions are highly concentrated near some value with a low variance and similar range. Therefore, the rank of calculated CVaR for different actions under $\eta = 0.5$ and $\eta = 0.1$ are still similar to the condition of $\eta = 1.0$. This shows that for short trips, risk-aware strategies do not make a difference.

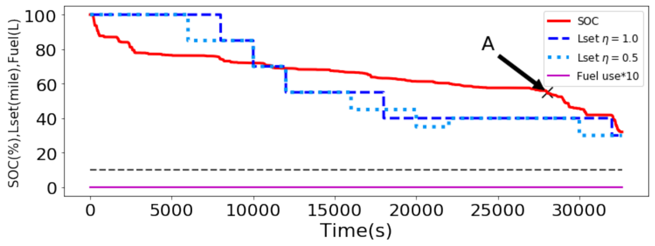

Fig. 10. Performance of the trained algorithm on a test trip with a distance of 35 miles with $\eta = 1.0$, $\eta = 0.5$ and $\eta = 0.1$. To make the figure clear, the L_{set} corresponding to $\eta = 0.1$ is not shown.

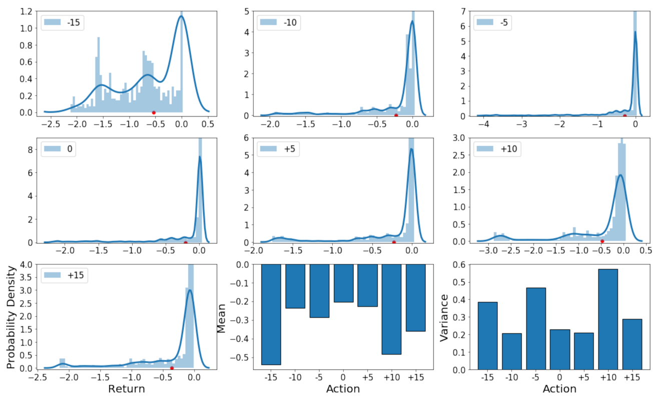

Fig. 11. Return distributions with mean and variance for all possible actions in state A. The smooth line is fitted with a kernel density estimator for better visualization, and the mean and variance were still calculated with the original data. The y-axis represents the probability density so that the value

could be higher than 1. The red dot represents the mean value of each distribution.

For a longer trip, as shown in Fig.12, the L_{set} changes significantly at each timestep under different η values, resulting in different fuel use and SOC curves. This can also be explained by observing the return distributions for some states. Fig. 13 shows the return distributions for state B. Compared with state A, it can be observed that the return distributions are much less concentrated and the probability densities for the peak are much lower than that of state A. Also, the range of the distributions is much wider on the negative side, indicating the possibilities of running out of battery. Therefore, for these wide and high variance distributions, the ranks of calculated CVaR under the condition of $\eta < 1$ are different from the standard risk-neutral condition. The lower the value of η, the more fuel would be used by the risk-aware algorithm.

The highest CVaR (the values corresponding to the selected action at each state) along the two delivery trips under the three values of η are shown in Fig. 14. As expected, for each state, the CVaR decreases as the value of η decreases from the upper figure, showing a more and more pessimistic estimation of performance. This can be easily explained by the definition of CVaR in (14). The reason why this seems not true for the last state of the lower figure is because the state information has become significantly different for the three η as different actions were taken during the task. For example, the final SOC and fuel consumption values are quite different for the three conditions. In addition, it can be observed that the values of CVaR are very similar at the beginning of the two trips. This can be explained by the fact that the two trajectories are not clearly distinguishable at first, but the discrepancy grows as they proceed. The exact value at the initial state reflects the algorithm's estimation of its performance based on the properties of the training trips and without knowing any information about the upcoming trip.

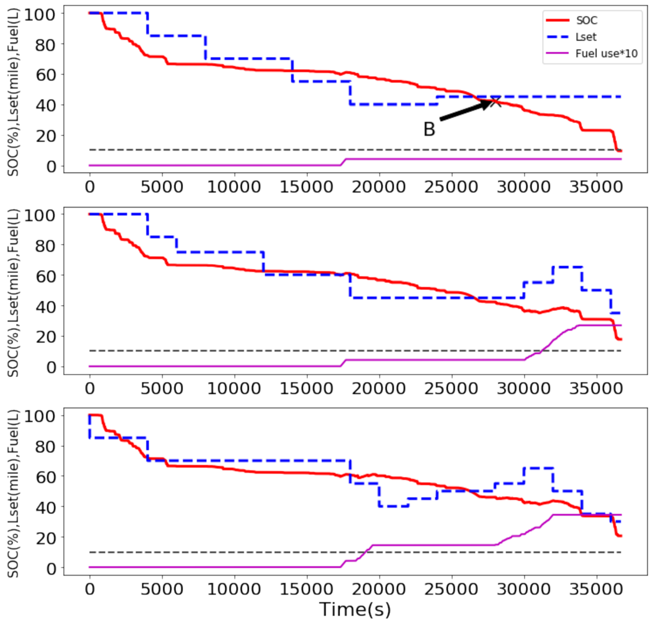

Fig. 12. Performance of the trained algorithm on a test trip with a distance of 48 miles with $\eta = 1.0$ (upper), $\eta = 0.5$ (middle) and $\eta = 0.1$ (lower).

Table II summarizes the average fuel use and average final SOC for the 58 test trips under the three η values. The lower the value of η, the more fuel was used to prevent the vehicle from running out of battery during the delivery trip, leading to a higher final SOC value. This is the tradeoff between fuel efficiency and the risk of running out of battery which could lead to delays of delivery tasks and dangerous conditions like stopping on highways. Consequently, after the algorithm is trained, η can be chosen according to different application scenarios and preferences of the users. For example, if the vehicle delivers in an urban area with frequent delivery stops, the η can be set close to 1 as there are enough charging opportunities and the energy intensities for this kind of driving is relatively low. However, if the delivery area includes long highway driving, a lower η might be used as the energy intensities for highway driving are usually high, and the charging rate of the range-extender is low compared with the energy consumption rate.

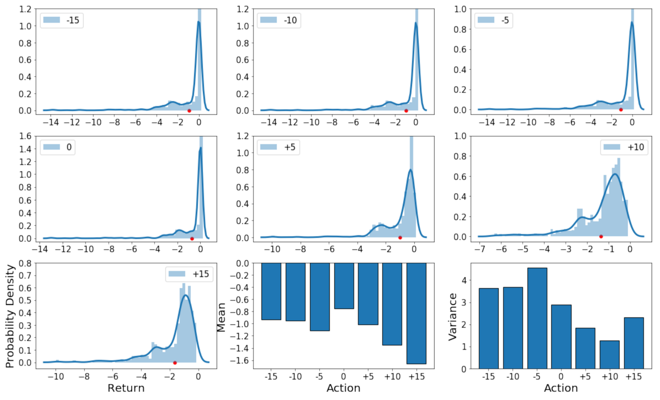

Fig. 13. Return distributions with mean and variance for all possible actions in state B.

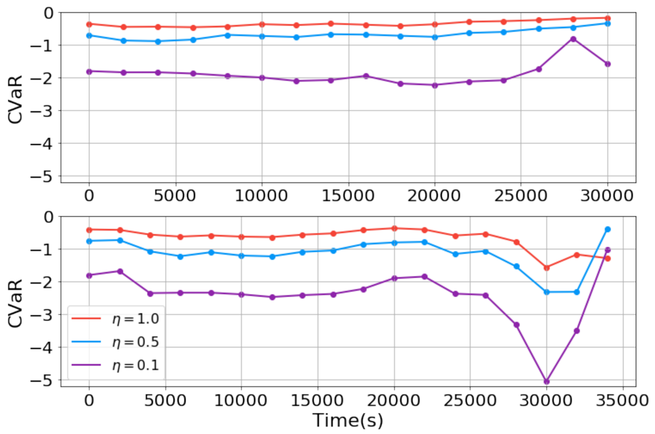

Fig. 14. Highest CVaR among possible actions along the two test trips.

TABLE II
TRADEOFF BETWEEN FUEL USE AND RISK OF RUNNING OUT OF BATTERY

	Average fuel use (L)	Average final SOC (%)
$\eta = 1.0$	0.72	18.6
$\eta = 0.5$	1.11	20.0
$\eta = 0.1$	1.94	23.1

VI. Discussion

A simple form of the original loss function is used in this work for faster training. According to the original loss function, for each transition pair (s_t, a_t, r_t, s_{t+1}), N samples of Z_{τ_i} and N' samples of $Z_{\tau'_j}$ should be sampled from the IQN and target IQN respectively. Then, the loss is calculated [11] as:

$$L(s_t, a_t, r_t, s_{t+1}) = \sum_{i=1}^{N} \frac{1}{N'} \sum_{j=1}^{N'} \left(\tau_i - \mathbb{1}_{\delta^{\tau_i, \tau'_j} < 0} \right) \delta^{\tau_i, \tau'_j}, \quad (16)$$

where:

$$\delta^{\tau_i, \tau'_j} = r_t + \gamma max_a Z_{\tau'_j}(s_{t+1}, a; \theta') - Z_{\tau_i}(s_t, a_t; \theta). \quad (17)$$

The loss for every transition pair in a sampled batch should be calculated as in (16) and then summed together. In our case, both N and N' were set to 1 so there was only one summation over n sampled transition pairs. A better return distribution estimation is expected to be achieved if more samples (N and N') are used during the training.

VII. Conclusion

Deep RL algorithms have proven to be powerful tools for solving ITS related problems. However, to implement these algorithms in real-world settings, the risk of large negative outcomes should be considered carefully as randomness and uncertainty are intrinsic characteristics of transportation systems. For applications involving risks and dangerous conditions, it is not enough for an algorithm to deliver good results on average. We adapt a risk-aware strategy based on IQN for managing the energy use in an EREV for delivery. We demonstrate the tradeoff between fuel efficiency and the risk of running out of battery power by changing the value of η in the risk measure of CVaR. This idea can be easily extended to other ITS related applications and would be especially beneficial for safety-critical problems like autonomous driving.

Acknowledgment

The information, data, or work presented herein was funded in part by the Advanced Research Projects Agency-Energy (ARPA-E) U.S. Department of Energy, under Award Number DE-AR0000795. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

References

[1] Lötjens, Björn, Michael Everett, and Jonathan P. How. "Safe reinforcement learning with model uncertainty estimates." In *2019 International Conference on Robotics and Automation (ICRA)*, pp. 8662-8668.

[2] K. Min, H. Kim and K. Huh, "Deep Distributional Reinforcement Learning Based High-Level Driving Policy Determination," in *IEEE Transactions on Intelligent Vehicles*, vol. 4, no. 3, pp. 416-424, Sept. 2019.

[3] Y. Du, W. ShangGuan, D. Rong and L. Chai, "RA-TSC: Learning Adaptive Traffic Signal Control Strategy via Deep Reinforcement Learning," *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, Auckland, New Zealand, 2019, pp. 3275-3280.

[4] X. Liang, X. Du, G. Wang and Z. Han, "A Deep Reinforcement Learning Network for Traffic Light Cycle Control," in *IEEE Transactions on Vehicular Technology*, vol. 68, no. 2, pp. 1243-1253, Feb. 2019.

[5] P. Wang, Y. Li, S. Shekhar and W. F. Northrop, "A Deep Reinforcement Learning Framework for Energy Management of Extended Range Electric Delivery Vehicles," *2019 IEEE Intelligent Vehicles Symposium (IV)*, Paris, France, 2019, pp. 1837-1842.

[6] P. Wang, Y. Li, S. Shekhar and W. F. Northrop, "Actor-Critic based Deep Reinforcement Learning Framework for Energy Management of Extended Range Electric Delivery Vehicles," *2019 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM)*, Hong Kong, China, 2019, pp. 1379-1384.

[7] Bertsekas, Dimitri P., and John N. Tsitsiklis. *Neuro-dynamic programming. Vol. 5.* Belmont, MA: Athena Scientific, 1996.

[8] R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction*. Cambridge MA: MIT Press, 2018.

[9] M. G. Bellemare, W. Dabney, and R. Munos, "A distributional perspective on reinforcement learning." In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 449-458. JMLR. org, 2017.

[10] Dabney, Will, Mark Rowland, Marc G. Bellemare, and Rémi Munos. "Distributional reinforcement learning with quantile regression." In *Thirty-Second AAAI Conference on Artificial Intelligence*. 2018.

[11] Dabney, Will, Georg Ostrovski, David Silver, and Rémi Munos. "Implicit quantile networks for distributional reinforcement learning." *arXiv preprint arXiv:1806.06923* (2018).

[12] A. Majumdar and M. Pavone, "How should a robot assess risk? Towards an axiomatic theory of risk in robotics," in Proc. Int. Symp. Robot. Res., 2017.

[13] Osband, Ian. "Risk versus uncertainty in deep learning: Bayes, bootstrap and the dangers of dropout." In *NIPS Workshop on Bayesian Deep Learning*. 2016.

[14] Gal, Yarin. "Uncertainty in deep learning." PhD diss., PhD thesis, University of Cambridge, 2016.

[15] M. J. Kochenderfer, *Decision Making Under Uncertainty: Theory and Application*. Cambridge, MA, USA: MIT Press, 2015.

[16] Pearce, Tim, Nicolas Anastassacos, Mohamed Zaki, and Andy Neely. "Bayesian Inference with Anchored Ensembles of Neural Networks, and Application to Exploration in Reinforcement Learning." *arXiv preprint arXiv:1805.11324* (2018).

[17] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, et al. Human-level control through deep reinforcement learning. Nature, 518(7540):529–533, 2015.

[18] C. Martinez, X. Hu, D. Cao, E. Velenis, B. Gao and M. Wellers, "Energy Management in Plug-in Hybrid Electric Vehicles: Recent Progress and a Connected Vehicles Perspective", *IEEE Transactions on Vehicular Technology*, vol. 66, no. 6, pp. 4534-4549, 2017.

[19] X. Qi, Y. Luo, G. Wu, K. Boriboonsomsin and M. J. Barth, "Deep reinforcement learning-based vehicle energy efficiency autonomous learning system," *2017 IEEE Intelligent Vehicles Symposium (IV)*, Los Angeles, CA, 2017, pp. 1228-1233.

[20] Z. Chen, L. Li, X. Hu, B. Yan and C. Yang, "Temporal-Difference Learning-Based Stochastic Energy Management for Plug-in Hybrid Electric Buses," in *IEEE Transactions on Intelligent Transportation Systems*.

[21] P. Wang, Y. Li, S. Shekhar and W. F. Northrop, "Uncertainty Estimation with Distributional Reinforcement Learning for Applications in Intelligent Transportation Systems: A Case Study," *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, Auckland, New Zealand, 2019, pp. 3822-3827.

[22] Nair, Vinod, and Geoffrey E. Hinton. "Rectified linear units improve restricted boltzmann machines." In *Proceedings of the 27th international conference on machine learning (ICML-10)*, pp. 807-814. 2010.

[23] Kingma, Diederik P., and Jimmy Ba. "Adam: A method for stochastic optimization." *arXiv preprint arXiv:1412.6980* (2014).