The Thirty-Seventh AAAI Conference on Artificial Intelligence (AAAI-23)

DARL: Distance-Aware Uncertainty Estimation for Offline Reinforcement Learning

Hongchang Zhang, Jianzhun Shao, Shuncheng He,Yuhang Jiang, Xiangyang Ji

Tsinghua University
hc-zhang19@mails.tsinghua.edu.cn

Abstract

To facilitate offline reinforcement learning, uncertainty estimation is commonly used to detect out-of-distribution data. By inspecting, we show that current explicit uncertainty estimators such as Monte Carlo Dropout and model ensemble are not competent to provide trustworthy uncertainty estimation in offline reinforcement learning. Accordingly, we propose a non-parametric distance-aware uncertainty estimator which is sensitive to the change in the input space for offline reinforcement learning. Based on our new estimator, adaptive truncated quantile critics are proposed to underestimate the out-of-distribution samples. We show that the proposed distance-aware uncertainty estimator is able to offer better uncertainty estimation compared to previous methods. Experimental results demonstrate that our proposed DARL method is competitive to the state-of-the-art methods in offline evaluation tasks.

Introduction

Offline reinforcement learning (Levine et al. 2020) has gained unprecedented attention recently, which is a remedy for costly, time-consuming, and unsafe online reinforcement learning (Dulac-Arnold, Mankowitz, and Hester 2019). However, applying traditional reinforcement learning methods directly to offline settings usually faces the challenge that the trained policy tends to diverge from the offline datasets due to the optimistic estimation of the target value of unseen state-action pairs (Munos 2003; Farahmand, Munos, and Szepesvári 2010; Scherrer et al. 2015; Fujimoto, Meger, and Precup 2019). This issue is commonly resolved by restricting the trained policy to in-distribution regions with low uncertainty (Kumar et al. 2019). Therefore, accurate uncertainty estimation is vital for offline reinforcement learning.

One way to tackle uncertainty estimation is to learn the data distribution directly and regard samples far from the distribution as uncertain. In offline reinforcement learning, variational autoencoder (VAE) (Kingma and Welling 2013) is commonly used to imitate the behavior policy (Fujimoto, Meger, and Precup 2019; Kumar et al. 2019). However, VAE cannot capture mixture distributions such as "medium-replay" dataset in D4RL benchmark (Fu et al. 2020). Other works choose to introduce explicit uncertainty estimators, such as Monte Carlo (MC) Dropout (Gal and Ghahramani 2016) and model ensemble (Lakshminarayanan, Pritzel, and Blundell 2016), into offline reinforcement learning. UWAC (Wu et al. 2021) applies MC Dropout and down-weights the samples with high uncertainty. However, the label in reinforcement learning keeps evolving, which differs from supervised learning and will introduce noise for uncertainty estimation. MOPO (Yu et al. 2020) defines the uncertainty estimator as the maximal standard deviation of an ensemble model. Nevertheless, (Liu et al. 2020a; Van Amersfoort et al. 2020) show that the deep ensemble model is not aware of the distances between unseen samples and training datasets, even in toy examples. Fig.1 demonstrates that VAE, MC Dropout, and model ensemble could not yield accurate uncertainty estimation, which corresponds to the distance between the data points for a low-dimensional offline reinforcement learning task. Note the dataset has more than one mode, which is common for offline reinforcement learning, such as mixed datasets in D4RL.

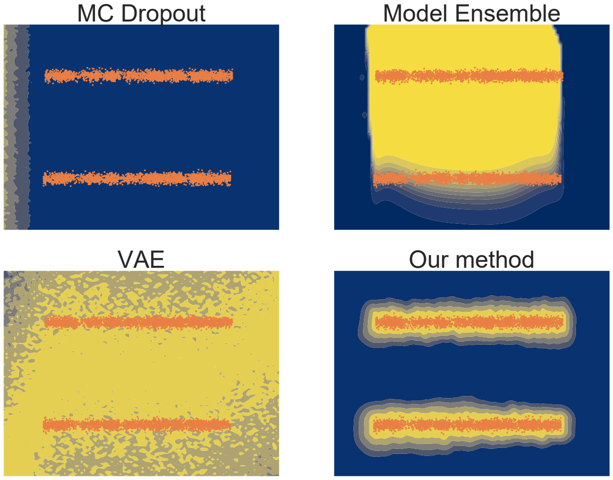

Figure 1: Visualization of uncertainty estimation for MC Dropout, model ensemble, VAE, and our method on a low-dimensional environment. The orange points denote the samples in the dataset. Yellow indicates low uncertainty, while blue indicates high uncertainty. X-axis and y-axis denote state and action, respectively.

These uncertainty estimators are parametric models which are targeted for reconstruction or regression objectives rather than directly tasked for uncertainty estimation, and therefore might discard important information such as the distances between different samples. What's more, parametric methods encourage in-distribution samples, but these samples might be out-of-sample (far from the dataset), leading to feature co-adaptation and poor performance (Kumar et al. 2021). In this paper, we resort to non-parametric models that preserve the data's mutual relations and propose distance-aware uncertainty estimation for offline reinforcement learning (DARL). The non-parametric method could tackle the multi-mode distribution. Also, it focuses on the sample-level distance and penalizes the out-of-sample actions. In particular, we devise a non-parametric particle-based cross entropy estimator which taps into k-nearest neighbor search. To make the k-nearest neighbor search feasible, we project the original data into a low-dimensional abstract representation space and restrict the Lipschitz constant of the mapping function to make distances in the feature space meaningful.

Based on this distance-aware uncertainty estimator, we further devise adaptive truncated quantile critics. Concretely, building upon a distributional critic (Dabney et al. 2018), we truncate the right tail of the Q-network by dropping several topmost atoms. The number of dropped atoms for a state-action pair correlates positively with the sample's uncertainty. In this way, our method tends to underestimate the uncertain samples. Our empirical evaluation shows that the proposed method is a better uncertainty estimator in offline reinforcement learning than previous methods. We also evaluate our proposed method on offline reinforcement learning benchmarks. The experimental results demonstrate that our proposed method outperforms the state-of-the-art methods on most tasks.

The contributions of this paper are as follows:

- We propose a non-parametric distance-aware uncertainty estimator and obtain excellent uncertainty in offline reinforcement learning.
- We devise adaptive truncated quantile critics based on the uncertainty estimator.
- We obtain competitive performance on offline reinforcement learning benchmarks.

Related Work

We will review previous uncertainty estimation works and uncertainty-based offline reinforcement learning methods in the following.

Uncertainty estimation. There is a large amount of research on estimating uncertainty in machine learning (Neal 2012; Blundell et al. 2015). Bayesian models (Mackay 1992) characterize the uncertainty estimation by posterior inference, such as MC Dropout which is scalable and simple to implement (Gal and Ghahramani 2016). Non-Bayesian deep ensemble (Lakshminarayanan, Pritzel, and Blundell 2016) maintains multiple neural networks initialized randomly and outperforms Bayesian models in practice (Ovadia et al. 2019). One drawback of MC Dropout and the deep ensemble model is that the hidden representation of the feature space does not reflect meaningful distances in the data space (Liu et al. 2020a; Van Amersfoort et al. 2020). Inspired from (Liu et al. 2020a), our method maps the high-dimensional input into a low-dimensional distance-preserving feature space. We devise a non-parametric cross entropy estimator for uncertainty estimation rather than rely on stochastic variational Gaussian process(SVGP) (Hensman, Matthews, and Ghahramani 2015) which only learns a bound of the true uncertainty.

Uncertainty-based offline reinforcement learning. Some works handle the uncertainty implicitly. Numerous researchers directly learn the data distribution and consider the samples far from the distribution as uncertain. (Liu et al. 2020b) directly estimates the density of the policy and reduces the update frequency of uncertain samples. However, this method requires an accurate estimation of the likelihood of the behavior policy, which is challenging in multi-dimensional scenarios (Van Oord, Kalchbrenner, and Kavukcuoglu 2016). BCQ (Fujimoto, Meger, and Precup 2019) and BEAR (Kumar et al. 2019) draw on VAE to imitate the behavior policy and train the policy to stay in the in-distribution region. Nevertheless, VAE suffers from its inability to capture mixture distributions. Another bunch of works do not resort to additional models to estimate the data distribution. CQL (Kumar et al. 2020) reduces the Q-value of samples with high uncertainty. Nevertheless, it is challenging to enumerate all uncertain samples. To underestimate the Q-values of the samples with high uncertainty, EDAC (An et al. 2021) maintains a large number of Q-networks and utilizes the ensemble's minimum to train the critics. However, the performance comes at the cost of a large ensemble.

Other methods rely on explicit uncertainty estimators. UWAC (Wu et al. 2021) introduces MC Dropout into offline reinforcement learning. However, UWAC's label is the unstable Q-value which is susceptible to the overestimation problem and incurs noise in uncertainty estimation. MOPO (Yu et al. 2020) trains an ensemble of the dynamics model and proposes an uncertainty-related penalty based on the model ensemble for imaginary samples. Nevertheless, the model ensemble will account for aleatory uncertainty for stochastic environments, which is unnecessary for offline reinforcement learning. Compared with these methods, our proposed uncertainty estimator is not affected by the noisy label and preserves the mutual relations among the data.

Background

We consider the standard reinforcement learning setting, which is always formalized with a Markov decision process (S, A, P, R, γ), with the state space S, the action space A, the transition function P, the reward function R, and the discount factor γ (Sutton and Barto 2018). In reinforcement learning, an agent starts from a state s and executes an action a at each time step. By interacting with the environment, the agent observes a next state s' and receives a reward r. The aim of reinforcement learning is to learn a policy $\pi(a|s)$ to receive accumulative rewards as much as possible: $\max_\pi \ \mathbb{E}[\sum_{t=0}^{\infty} \gamma^t r_t]$. In the rest of the paper, we also use $\pi(s)$ to denote the policy.

Distributional reinforcement learning. Distributional reinforcement learning (Bellemare, Dabney, and Munos

2017) aims to learn the distribution of cumulative reward: $Z_\pi(s,a) := \sum_{t=0}^{\infty}\gamma^t r_t$, where $s_0 = s, a_0 = a$, and $a_t \sim \pi(\cdot|s_t)$.

QR-DQN (Dabney et al. 2018) approximates the distribution $Z_\pi(s,a)$ with $\frac{1}{N_A}\sum_{i=1}^{N_A}\delta\left(\theta^i(s,a)\right)$, which is a mixture of atoms-Dirac delta functions at locations $\theta^1(s,a),...,\theta^{N_A}(s,a)$, where $\theta(\cdot,\cdot)$ is a parametric model and N_A is the number of quantiles. The parameter for $\theta(\cdot,\cdot)$ is updated by minimizing the 1-Wasserstein distance between $Z_\pi(s,a)$ and the temporal target: min $W_1(Z_\pi(s,a), r + \gamma Z_\pi(s',a'))$, where $a' \sim \pi(\cdot|s')$ and $W_1(\cdot,\cdot)$ is the 1-Wasserstein distance between two distributions.

Offline reinforcement learning. In offline reinforcement learning, a dataset $\mathcal{D}$ is provided. We denote the behavior policy which produces the dataset as π_β. The state-action distribution for a policy π is defined as $\mu(s,a) = \frac{\sum_{t=0}^{\infty}\gamma^t\mu_t(s,a)}{\sum_{t=0}^{\infty}\gamma^t}$, where $\mu_t(s,a)$ is the density of (s,a) at timestep t. The state-action distribution for the dataset is denoted by μ_β, accordingly.

Offline reinforcement learning suffers from the out-of-distribution challenge. The standard reinforcement learning algorithms might derive a policy that generates out-of-distribution actions in the offline setting. They have no chance to adjust the real Q-values of these actions correctly.

TD3BC (Fujimoto and Gu 2021) builds on top of TD3 (Fujimoto, Hoof, and Meger 2018). It mainly focuses on plugging a behavior cloning regularization term to the policy training in TD3. The policy loss is

$$\pi = \underset{\pi}{\operatorname{argmax}}\mathbb{E}_{(s,a)\sim\mathcal{D}}\left[vQ(s,\pi(s)) - (\pi(s)-a)^2\right], \quad (1)$$

where the hyperparameter v is used to make a balance between reinforcement learning and imitation learning objectives. To make these two terms in the same range, v is defined as $v = \frac{2.5}{\frac{1}{N}\sum_{(s_i,a_i)}|Q(s_i,a_i)|}$, where N is the size of a mini-batch.

Particle-based entropy estimator. Particle-based entropy estimator is an unbiased and non-parametric estimator of entropy (Singh et al. 2003). For a random variable X with dimension l, n datapoints $\{x_i\}_{i=1}^{n}$ are sampled from its distribution. The approximated entropy is defined as $H(X) \propto \sum_i^n \log c_i^k$, where c_i^k is the volume of a hypersphere with the center x_i. The radius is the distance between x_i and its k-th nearest neighbor x_i^k. The entropy estimator can be rewritten as $H(X) \propto \sum_i^n \log ||x_i - x_i^k||^l$. Note that the k-nearest search is computationally inefficient when provided with a large dataset.

Method

Our method comprises two parts: distance-aware uncertainty estimator and adaptive truncated quantile critic. The distance-aware uncertainty estimator is the prerequisite for the latter part. We will introduce these two components in detail in the following and describe the implementation details afterwards.

Distance-Aware Uncertainty Estimator

We characterize the uncertainty for the trained policy as its cross entropy against the behavior policy: $H(\mu,\mu_\beta) = -\sum_{s,a}\mu(s,a)\log\mu_\beta(s,a)$. The $H(\mu,\mu_\beta)$ is a better uncertainty estimator than $H(\mu_\beta,\mu) = -\sum_{s,a}\mu_\beta(s,a)\log\mu(s,a)$ for the reason that the latter one provides no uncertainty when $\mu(s,a) > 0$ and $\mu_\beta(s,a) = 0$ for some (s,a). On the contrary, the former one offers high uncertainty when μ deviates from the support of μ_β. To simplify the notation, x and y represent the state-action pairs sampled from $\mu_\beta(s,a)$ and $\mu(s,a)$, respectively.

However, it is challenging to estimate $H(\mu,\mu_\beta)$ due to the unavailable density μ_β. We derive a particle-based cross-entropy estimator based on the non-parametric entropy estimator (Singh et al. 2003): $\hat{H}(\mu,\mu_\beta) = -\frac{1}{n}\sum_{i=1}^{n}\log\hat{\mu}_\beta(y_i)$, where $y_i \sim \mu$, n is the number of samples drawn from μ, and $\hat{\mu}_\beta(y_i)$ is an estimator of $\mu_\beta(y_i)$. Given a dataset $\{x_j\}_{j=1}^{m}$ drawn form μ_β with the size m, the $\hat{\mu}_\beta(y_i)$ can be written as $\hat{\mu}_\beta(y_i) = \frac{k\Gamma(l/2+1)}{m\pi^{l/2}R_{i,k,m}^l}$, where l is the dimension of the random variable and $R_{i,k,m}$ is the Euclidean distance of y_i to its closest k-th neighbor in $\{x_j\}_{j=1}^{m}$. Note that $\frac{\pi^{l/2}R_{i,k,m}^l}{\Gamma(l/2+1)}$ is the volume of the sphere of the radius $R_{i,k,m}$. We can obtain the estimator of cross entropy $H(\mu,\mu_\beta)$ by considering the distance between each particle sampled from μ and its k-th closest neighbor sampled from μ_β:

$$\hat{H}(\mu,\mu_\beta) = -\frac{1}{n}\sum_{i=1}^{n}\log\frac{k\Gamma(l/2+1)}{m\pi^{l/2}R_{i,k,m}^l} \propto \frac{1}{n}\sum_{i=1}^{n}\log R_{i,k,m}^l. \quad (2)$$

Based on the uncertainty for the distribution μ, we can also derive the uncertainty estimator for a given test sample $\hat{y}$ by considering the μ distribution as taking the form of the delta distribution: : $q(x) = \infty$ when $x = \hat{x}$, otherwise 0. The uncertainty of the test sample is defined as :

$$U(\hat{y}) = \log||\hat{y} - x_j^k||_{\mathcal{X}}^l, \quad (3)$$

where x_j^k is the test sample's k-th nearest neighbor in $\{x_j\}_{j=1}^{m}$ and $||\cdot||_{\mathcal{X}}$ is the norm in $\mathcal{X}$.

We show that the proposed uncertainty estimator is closely related to the theoretical study on provable efficiency in offline RL where ξ-uncertainty quantifier is important in the analysis (Xie et al. 2021).

Definition 1. (ξ-Uncertainty Quantifier (Jin, Yang, and Wang 2021)). For a series of penalty $\{\Gamma_t\}_{t=1}^{T}$ ($\Gamma_t : S \times A \to \mathbb{R}$), if it holds that $P(|\mathcal{B}Q_{t+1}(s,a) - \hat{\mathcal{B}}Q_{t+1}(s,a)| \leq \Gamma_t(s,a)) \geq 1-\xi$ for all $(s,a) \in S \times A$, where $\mathcal{B}$ is the Bellman operator and $\hat{\mathcal{B}}$ is the empirical Bellman operator, then we say the penalty $\{\Gamma_t\}_{t=1}^{T}$ is a ξ-uncertainty quantifier.

The generally used LCB-penalty is a ξ-uncertainty quantifier. Based on the proposed uncertainty estimator, we have the following theorem.

Theorem 2. *For a linear MDP where the feature function for a given (s,a) is $\phi(\cdot,\cdot) : S \times A \to \mathbb{R}^l$, the k-nearest neighbor uncertainty $U(s,a) = \beta_t\log||\phi(s,a) - \phi(s_j,a_j)^k||^l$ is an estimation to a provably efficient LCB-penalty, which takes the form $\Gamma(s,a) = \beta_t\left[\phi(s,a)^\top\Lambda^{-1}\phi(s,a)\right]^{1/2}$, where*

$\Lambda=\sum_{j=1}^{m} \phi\left(s_{j}, a_{j}\right) \phi\left(s_{j}, a_{j}\right)^{\top}$ and β_t is an appropriately selected parameter.

Theorem 2 shows that penalizing the state-action value function based on the k-nearest neighbor uncertainty estimation yields an efficient offline RL algorithm for linear MDPs. We denote the optimal policy and the policy induced by the penalized state-action value function as π^* and $\bar{\pi}$, respectively. We can obtain the gap between the expected cumulative rewards under the two policies.

Corollary 3. *Under the Theorem 2's condition, it holds that $J^{\pi^*} - J^{\bar{\pi}} \leq \sum_{t=1}^{T} \gamma^t \mathbb{E}_{\pi^*}[\Gamma_t(s_t, a_t)]$, where J^{π^*} and $J^{\bar{\pi}}$ are expected cumulative rewards under policy π^* and $\bar{\pi}$, respectively.*

Corollary 3 shows the optimality gap brought by the pessimistic value iteration. Note that it is information-theoretically optimal for linear MDPs.

High Dimensional Uncertainty Estimation

For high-dimensional observation and action space, k-nearest search is computationally heavy when the number of samples is over one million. KD-tree is a data structure which could reduce the computational complexity to $\log(m)$. However, k-nearest search based on KD-tree is not scalable in deep reinforcement learning for the reason that the searching in the original data space is more time-consuming than gradient backpropagation. Therefore, we utilize a deep neural network to process the original state-action pairs into low-dimensional feature space to accelerate the search.

Rather than utilizing the hidden representation of the Q-network or a learned dynamics model, we choose a randomized neural network as a feature extractor. The reason is that the hidden representations for these tasks are changing during the training process, which makes the searching infeasible for an established KD-tree. In addition, the hidden representation is always high-dimensional, which makes the searching computationally expensive in practice. What's more, specific objectives might discard important information in the dataset, such as the relations among data points.

To make the randomized neural network preserve such relations in the original space and avoid feature collapse (Van Amersfoort et al. 2020), we expect the hidden mapping $h(x)$ to be sensitive to the changes in the input. To this end, we must ensure that the distance in the hidden space $||h(x) - h(x')||_{\mathcal{H}}$ corresponds to the distance in the input space $||x - x'||_{\mathcal{X}}$. We build the distance-preserving randomized neural network upon the idea of bounding the Lipschitz constant of each layer. Similar to (Liu et al. 2020a), for a D-layer network h, we build each layer h_d as a residual block $h_d(x) = x + g_d(x)$, and bound the nonlinear mapping g_d's Lipschitz constant to be less than α by spectral normalization (Miyato et al. 2018). To be specific, for $g_d(x) = \text{relu}(W_d x + b_d)$, we estimate the spectral norm of W_d as $\sigma(W_d)$ according to the power iteration method and normalize W_d by $W_d = u * W_d/\sigma(W_d)$ when $u < \sigma(W_d)$, where u is a hyperparameter which controls the upper bound on $||W_d||_2$. In this way, the Lipschitz constant of h is bounded in $[(1-\alpha)^{D-1}, (1+\alpha)^{D-1}]$. The proof can be found in Liu et al. (2020a).

We now put it all together to introduce our distance-aware uncertainty estimator. For samples $\{y_i\}_{i=1}^n$ generated from the distribution μ, we project the samples using the aforementioned distance-preserving randomized neural network $h(\cdot)$ to get features $\{h(y_i)\}_{i=1}^n$. The uncertainty for μ given dataset $\{x_j\}_{j=1}^m$ sampled from π_β is $\sum_{i=1}^n \frac{1}{n} \log ||h(y_i) - h(x_j)^k||_{\mathcal{H}}^{l_{\mathcal{H}}}$, where $h(x_j)^k$ is the k-th nearest neighbor for $h(y_i)$ in the feature space and $l_{\mathcal{H}}$ is the dimension of the feature space.

Adaptive Truncated Quantile Critics

Based on the proposed uncertainty estimator, it is natural to penalize the target value by the uncertainty: $\hat{\mathcal{B}}Q(s,a) - U(s', a')$. From a distributional viewpoint, we propose adaptive truncated quantile critics which underestimates the samples with high uncertainty to handle the out-of-distribution issue.

The adaptive truncated quantile critics is built upon the distributional algorithm TQC(Truncated Quantile Critics) (Kuznetsov et al. 2020) that alleviates overestimation in policy evaluation. TQC maintains an ensemble of distributional critics and mixes all the atoms to form a mixture distribution. Then it drops several largest atoms and averages remaining atoms to obtain a truncated mixture return distribution.

For the offline setting, the issue of overestimation will exacerbate because of the out-of-distribution target (Fujimoto, Meger, and Precup 2019). Samples with different uncertainty should be underestimated at different levels. We utilize the distance-aware uncertainty estimator to build adaptive truncated quantile critics.

We train N_C distributional critics $Z_1, ..., Z_{N_C}$. For each distributional critic, we learn N_A quantiles. The target value of state-action pair (s', a') for critic Z_c is defined by $Z_c(s,a) := \frac{1}{N_A} \sum_{i=1}^{N_A} \delta\left(\theta_c^i(s', a')\right)$. We aggregate the atoms of all the critics to form a set $\{\theta_c^i(s', a') | i = 1, ..., N_A, c = 1, ..., N_C\}$.

Then we sort the elements in the set in ascending order, and the sorted set is $\{\hat{\theta}_i\}_{i=1}^{N_A \times N_C}$. We drop $f(U(s', a'))$ largest atoms to form an adaptive truncated target. The function $f(.)$ positively relates to the uncertainty. When the uncertainty of (s', a') is high, we drop a large portion of atoms to underestimate its expected return. The target distribution is defined as $Y(s', a') := \sum_{i=1}^{N'} \frac{1}{N'} \delta(\hat{\theta}_i(s', a'))$, where $N' = N_A \times N_C - f(U(s', a'))$ is the number of remained atoms. The loss function for the critics is

$$\mathcal{L} = \frac{1}{N' N_A} \sum_{j=1}^{N_A} \sum_{i=1}^{N'} \rho_{\tau_i}(r + \gamma \hat{\theta}_i(s', a') - \theta_j(s,a)), \quad (4)$$

where $\rho_\tau(z) = z(\tau - \mathbb{I}(z < 0)), \forall z \in \mathbb{R}$ at τ-quantile.

Compared to TQC, which drops identical atoms for all the state-action pairs, our method sets the truncation level according to the target's uncertainty, which is more flexible and more applicable in offline settings.

```
Algorithm 1: DARL
Input: offline dataset $\mathcal{D}$, update iterations $t_{max}$, $k$ for $k$-nearest search, the size of the dataset $m$
Parameter: randomized neural network $h$, policy network $\pi$, an ensemble of critics $Z_1, ..., Z_{N_C}$
Output: learned policy network $\pi$
Initialize the policy network, distributional critics and the randomized neural network
Project the whole dataset to $h$ to get the features $\{h(s_i, a_i)\}_{i=1}^{m}$
Build a KD-tree for all the features
for $t = 1, 2, \ldots, t_{max}$ do
    Sample a mini-batch of samples $(s, a, r, s')$ from $\mathcal{D}$
    Calculate the uncertainty for the target $U(s', \pi(s'))$
    Formulate the truncated target
    Update the distributional critics according to Eq.4
    Update the policy network according to Eq.1
end for
```

Implementation Details

For the distance-aware uncertainty estimator, we use a two-layer randomized neural network to project a state-action pair into the feature space. The output dimension is set to be less than 10 to accelerate the k-nearest search. For a given dataset, we build a KD-tree on the whole dataset other than conduct k-nearest search in a minibatch (Liu and Abbeel 2021) to ensure accurate uncertainty estimation. The practical time consumption of building and searching for KD-tree in the feature space is slight compared to the training process. The specific cost is described in the experiment.

For the training of the critics, we maintain a set of distributional critics and learn multiple quantiles for each critic. The quantile drop function $f(U(s', a'))$ is a truncated linear function: $f(U(s', a')) = \text{clip}(\eta * U(s', a')), C_{\text{min}}, C_{\text{max}})$, where η is a hyperparameter. C_{min} and C_{max} are the minimum and maximum number of clipped quantiles, respectively.

For the training of the actor, we adopt the training procedure of TD3BC, which helps the regularization when the data distribution is narrow. The algorithm is described in Alg. 1

Experiment

In this section, we conduct several experiments to justify the validity of our proposed method. We aim to answer three questions: (1) Is our proposed distance-aware uncertainty estimator better than previous uncertainty estimators? (2) Does our method perform better on standard offline benchmarks than previous methods? (3) How does the adaptive truncated critics contribute to our method?

Uncertainty Estimation

We show the behavior of the proposed uncertainty estimator in a low dimensional RL environment, out-of-detection benchmarks-KDDCUP and Thyroid datasets, and D4RL benchmark.

Low Dimensional RL Environment. To visualize the uncertainty estimation clearly, we devise an environment where the dimension of the continuous state space is 1 with the range $[-100, 100]$. The action space is one-dimensional with the range $[-1.2, 1.2]$. The action denotes the agent's movement at each time step. If the agent moves right, it gets a reward of $+1$. Otherwise, it gets -1. The maximum episode length is set to 100. We create a well-performed policy $\pi_1(a|s) = 1 + 0.1 * \mathcal{N}(0, 1)$, where $\mathcal{N}(0, 1)$ is a Gaussian distribution. It receives positive rewards with a high probability. A relatively bad policy $\pi_2(a|s) = -1 + 0.1 * \mathcal{N}(0, 1)$, which always receives negative rewards is also created. At each time step, we sample an action randomly from these two polices and collect $20,000$ samples to formulate a fixed dataset.

We compare our methods to MC Dropout, model ensemble, and VAE. The details of implementation of these methods are described in Appendix. The results are shown in Fig. 1. The x-axis denotes the state, and the y-axis denotes the action. We normalize the dataset's state with mean 0 and standard deviation 1. The result shows that MC Dropout could not distinguish between in-distribution and out-of-distribution regions. It tends to produce a similar uncertainty estimation for most state-action pairs. The reason might be that the MC Dropout's label is the Q-value function, which is unstable during the training process. The model ensemble is able to recognize the out-of-distribution states. Nevertheless, it predicts the out-of-distribution actions incorrectly. The VAE could produce a reasonable uncertainty prediction for the region where the action is larger than 1.1 or smaller than -1.1. However, VAE could not estimate the uncertainty correctly for the out-of-distribution region where the action range is $[-0.9, 0.9]$. The reason might be that the latent variable distribution is unimodal and could not capture multi-modal mixture datasets. Compared to these methods, our method shows low uncertainty near the dataset and high uncertainty in the region far from the dataset. As we can see, our method is a better uncertainty estimator.

We also compare with standard deep ensemble's uncertainty which is commonly used in computer vision (Lakshminarayanan, Pritzel, and Blundell 2016), and DUE which also uses spectral normalization to estimate uncertainty (van Amersfoort et al. 2021). The result is shown in Fig. A1 in Appendix. DUE outputs a Gaussian distribution which cannot capture multi-mode distribution. The standard ensemble cannot distinguish the OOD region.

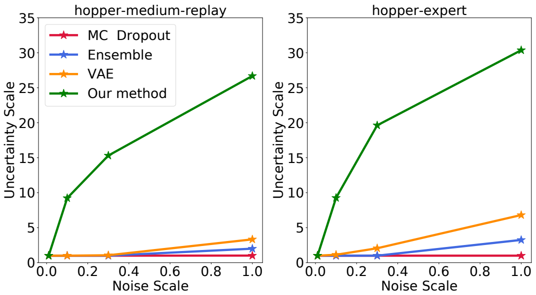

Figure 2: Uncertainty estimation results for MC Dropout, model ensemble, VAE, and our method.

KDDCUP	Precision	Recall	F1
OC-SVM	0.7457	0.8523	0.7954
DCN	0.7696	0.7829	0.7762
DSEBM	0.8619	0.6446	0.7399
DAGMM	0.9297	0.9442	0.9369
MemAE	0.9627	0.9655	0.9641
GOAD	-	-	**0.9840**
Our Method	**0.9787**	**0.9965**	**0.9875**

Table 1: Results on KDDCUP dataset

Thyroid	Precision	Recall	F1
OC-SVM	0.3639	0.4239	0.3887
DCN	0.3319	0.3196	0.3251
DSEBM	0.1319	0.1319	0.1319
DAGMM	0.4766	0.4834	0.4782
GOAD	-	-	**0.745**
Our Method	0.6132	0.8935	**0.7273**

Table 2: Results on Thyroid dataset

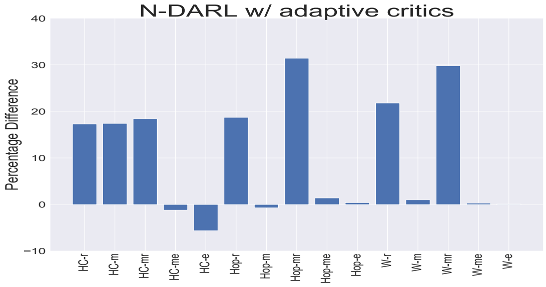

Figure 3: Percentage difference of the performance of an ablation of our proposed method, compared to N-DARL. N-DARL w/ adaptive critics refers to the N-DARL with the adaptive truncated quantile critics.

Out-of-detection benchmarks. we compare with OC-SVM (Chen, Zhou, and Huang 2001), DCN (Yang et al. 2017), DSEBM (Zhai et al. 2016), DAGMM (Zong et al. 2018), MemAE (Gong et al. 2019), and GOAD (Bergman and Hoshen 2020) on commonly used out-of-detection benchmarks KDDCUP and Thyroid datasets. When our method performs on the dataset, the datapoints which have larger uncertainty than a predefined threshold will be marked as anomalies. We take the anomaly class as positive, and define precision, recall, and F1 score accordingly. Table 1 and Table 2 report the average precision, recall, and F1 score to compare anomaly detection performance. The results show that the proposed estimator is comparable to state-of-the-art OOD detection methods.

D4RL dataset. We also test our method's validity in uncertainty estimation on "hopper-expert-v2" and "hopper-medium-replay" datasets in D4RL. The implementation of our uncertainty estimator, MC Dropout, the model ensemble is similar to the former low-dimensional setting. For each (s, a) in the dataset, we create out-of-distribution samples by adding a noise to the action $\hat{a} = a + b_i * \mathcal{N}(0, 1)$. We create 4 out-of-distribution datasets by setting b_i in $\{0.01, 0.1, 0.3, 1\}$.

These methods have distinct definitions for uncertainty estimation and are not comparable numerically. We regard the median uncertainty as 1 for the dataset with $b_i = 0.01$ for each method and rescale the median uncertainty for other datasets accordingly. The experimental results of all methods are summarized in Fig. 2. (The results of the original scale of all methods are shown in Appendix Table A7. As the results show, MC Dropout produces approximately similar uncertainty predictions for in-distribution and out-of-distribution samples. The model ensemble also cannot detect out-of-distribution samples in the offline dataset when the noise scale is small. Since the behavior policy for "hopper-expert" has only one mode, VAE can recognize out-of-distribution samples. However, VAE would struggle when trained on mixed datasets. Compared to these methods, our proposed method has better uncertainty estimation and can spot out-of-distribution samples when the noise scale is only 0.1.

Performance on Offline Mujoco Datasets

To evaluate the validity of DARL in high-dimensional settings, we perform experiments using the Mujoco control suites in D4RL's "-v2" benchmarks (Fu et al. 2020). The experimental results of the previous methods and DARL are summarized in Table 3. Since TD3BC (Fujimoto and Gu 2021) experiments on "-v0" dataset, we rerun TD3BC on "-v2" environments for fair comparisons.

As shown in Table 3, DARL outperforms or achieves competitive performance compared to the state-of-the-art methods on most tasks. Especially, our proposed method exceeds the previous methods on "random" and "medium-replay" datasets. The experiments demonstrate that uncertainty estimation is vital for offline learning. For DARL, the time complexity of building a KD-tree for a dataset is less than one minute. The time cost for k-nearest search for a mini-batch is less than the gradient calculation and backpropagation. We also test the runtime of our proposed algorithm. The runtime of f CQL is 31.5 epoch per second while DARL is 28.8 epoch per second, which is comparable to CQL.

Performance on Adroit Hand Task

Adroit hand manipulation tasks are more complicated than Mujoco control tasks. These tasks aim to control a 24-DoF robotic hand to execute specific options. The Adroit tasks consist of hammering a nail, opening a door, twirling a pen, or picking up and moving a ball. We experiment on two types of datasets for each environment: "human" and "cloned". The results of the methods are summarized in Table 4. Our method is competitive to previous methods. Especially, for pen-human task, our method outperforms the previous methods by a large margin.

Ablation Study

We perform an ablation study over the components in our method. Fig. 3 shows the result of the effectiveness of the

Task Name	BEAR	BRAC	UWAC	TD3BC	CQL	IQL	DARL
HC-r	12.6 ± 1.0	24.3 ± 0.7	2.3 ± 0.0	11.7 ± 1.3	31.3 ± 3.5	-	**32.4 ± 2.2**
Hc-m	42.8 ± 0.1	51.9 ± 0.3	43.7 ± 0.4	48.8 ± 0.2	46.9 ± 0.4	47.4 ± 0.2	**69.8 ± 3.8**
HC-e	92.6 ± 0.6	39.0 ± 13.8	94.7 ± 1.1	93.4 ± 4.2	97.3 ± 1.1	-	**98.9 ± 4.0**
HC-me	45.7 ± 4.2	52.3 ± 0.1	47.0 ± 6.0	87.3 ± 1.8	95.0 ± 1.4	86.7 ± 5.3	95.7 ± 3.6
HC-mr	39.4 ± 0.8	48.6 ± 0.4	38.9 ± 1.1	44.8 ± 0.3	45.3 ± 0.3	44.2 ± 1.2	**59.6 ± 3.3**
Hop-r	3.6 ± 3.6	8.1 ± 0.6	2.6 ± 0.3	8.3 ± 0.4	5.3 ± 0.6	-	**32.3 ± 2.9**
Hop-m	55.3 ± 3.2	77.8 ± 6.1	52.6 ± 4.0	58.5 ± 0.7	61.9 ± 6.4	66.2 ± 5.7	63.7 ± 2.6
Hop-e	39.4 ± 20.5	78.1 ± 52.3	111.0 ± 0.8	110.1 ± 0.1	106.5 ± 9.1	-	108 ± 1.3
Hop-me	66.2 ± 8.5	81.3 ± 8.0	54.8 ± 3.2	99.5 ± 5.1	96.9 ± 15.1	91.5 ± 14.3	**110.6 ± 0.7**
Hop-mr	57.7 ± 16.5	62.7 ± 30.4	31.1 ± 14.8	60.3 ± 13.7	86.3 ± 7.3	94.7 ± 8.6	**96.7 ± 6.9**
W-r	4.3 ± 1.2	1.3 ± 1.4	1.5 ± 0.3	0.8 ± 0.5	5.4 ± 1.7	-	**21.7 ± 2.7**
W-m	59.8 ± 40.0	59.7 ± 39.9	66.0 ± 9.0	84.2 ± 0.2	79.5 ± 3.2	78.3 ± 8.7	84.5 ± 0.2
W-e	110.1 ± 0.6	55.2 ± 62.2	108.4 ± 0.5	110.9 ±0.2	109.3 ± 0.1	-	**111.2 ± 0.4**
W-me	107.0 ± 2.9	9.3 ± 18.9	85.7 ± 14.0	110.4 ± 0.2	109.1 ± 0.2	109.6 ± 1.0	110 ± 0.6
W-mr	12.2 ± 4.7	40.1 ± 47.9	27.1 ± 9.6	73.7 ± 7.4	76.8 ± 10.0	73.8 ± 7.1	**99.4 ± 4.1**

Table 3: Results of BEAR, BRAC, UWAC, TD3BC, CQL, IQL(Kostrikov, Nair, and Levine 2022), and DARL on the offline Mujoco tasks. The results are averaged over six seeds. HC = HalfCheetah, Hop = Hopper, W = Walker, r = random, m = medium, mr = medium-replay, me = medium-expert, e = expert.

Task Name	BC	BEAR	CQL	DARL
pen-human	25.8	-1.0	55.8	70.4 ± 8.2
hammer-human	3.1	0.3	2.1	3.2 ± 1.1
door-human	2.8	-0.3	9.1	2.3 ± 0.3
relocate-human	0.0	-0.3	0.35	0.2 ± 0.0
pen-cloned	38.3	26.5	40.3	41.6 ± 4.7
hammer-cloned	0.7	0.3	5.7	0.4 ± 0.1
door-cloned	0.0	-0.1	3.5	3.5 ± 0.7
relocate-cloned	0.1	-0.3	-0.1	0.0 ± 0.0

Table 4: Normalized scores of all methods on Adroit domains, averaged across six seeds.

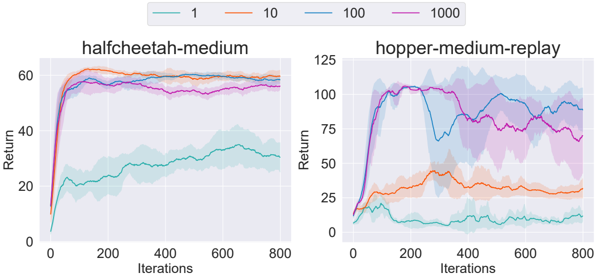

Figure 4: Varying the scale of η. Mean and standard deviation are plotted over five seeds.

proposed adaptive truncated quantile critics compared to a naive version of DARL without the adaptive component. We call the naive version "N-DARL". Note that N-DARL's policy loss is similar to TD3BC and only replaces the Q-function with a distributional ensemble. Since N-DARL has already achieved expert-level performance on "medium-expert" and "expert" tasks, it is hard for our two variants to perform better on these tasks. For other tasks, the result shows that adaptive truncated quantile critics is an essential component to achieve strong performance.

Adaptive truncated quantile critics. We now study the relation of our method with the hyperparameter η which controls the sensitiveness of the target value function to the out-of-distribution samples. In Fig. 4 , we test different values for hyperparameter η in $\{1, 10, 100, 1000\}$ on "halfcheetah-medium" and "hopper-medium-replay". Note that $\eta = 1000$ approximately corresponds to truncating a large amount atoms for all in-distribution and out-of-distribution samples while $\eta = 1$ approximately corresponds to not underestimating all samples. The result shows that when η is large, the value function of in-distribution samples will be underestimated, and the performance deteriorates. When η is small, out-of-distribution samples will be overestimated and will hurt the performance. Therefore, appropriate truncation is important for proper underestimation.

We also test the impact of value k on our method. The result is shown in Table A6 in Appendix.

Conclusion

In this paper, we propose a distance-aware uncertainty estimator for offline reinforcement learning. We introduce adaptive truncated quantile critics to underestimate the samples with high uncertainty. We show that the proposed distance-aware uncertainty estimator can offer better uncertainty estimation than previous methods. Experimental results demonstrate that our proposed DARL method outperforms the state-of-the-art methods in offline reinforcement learning evaluation tasks. For future work, we will try to find better uncertainty estimators to apply to offline settings.

Acknowledgements

This work was supported by the National Key R&D Program of China under Grant 2018AAA0102800, National Natural Science Foundation of China under Grant 61620106005.

References

An, G.; Moon, S.; Kim, J.-H.; and Song, H. O. 2021. Uncertainty-based offline reinforcement learning with diversified q-ensemble. *Advances in Neural Information Processing Systems*, 34.

Bellemare, M. G.; Dabney, W.; and Munos, R. 2017. A distributional perspective on reinforcement learning. In *International Conference on Machine Learning*, 449–458. PMLR.

Bergman, L.; and Hoshen, Y. 2020. Classification-Based Anomaly Detection for General Data. In *International Conference on Learning Representations*.

Blundell, C.; Cornebise, J.; Kavukcuoglu, K.; and Wierstra, D. 2015. Weight uncertainty in neural network. In *International Conference on Machine Learning*, 1613–1622. PMLR.

Chen, Y.; Zhou, X. S.; and Huang, T. S. 2001. One-class SVM for learning in image retrieval. In *Proceedings 2001 International Conference on Image Processing (Cat. No. 01CH37205)*, volume 1, 34–37. IEEE.

Dabney, W.; Ostrovski, G.; Silver, D.; and Munos, R. 2018. Implicit quantile networks for distributional reinforcement learning. In *International conference on machine learning*, 1096–1105. PMLR.

Dulac-Arnold, G.; Mankowitz, D.; and Hester, T. 2019. Challenges of real-world reinforcement learning. *arXiv preprint arXiv:1904.12901*.

Farahmand, A. M.; Munos, R.; and Szepesvári, C. 2010. Error propagation for approximate policy and value iteration. In *Advances in Neural Information Processing Systems*.

Fu, J.; Kumar, A.; Nachum, O.; Tucker, G.; and Levine, S. 2020. D4rl: Datasets for deep data-driven reinforcement learning. *arXiv preprint arXiv:2004.07219*.

Fujimoto, S.; and Gu, S. S. 2021. A Minimalist Approach to Offline Reinforcement Learning. *arXiv preprint arXiv:2106.06860*.

Fujimoto, S.; Hoof, H.; and Meger, D. 2018. Addressing function approximation error in actor-critic methods. In *International Conference on Machine Learning*, 1587–1596. PMLR.

Fujimoto, S.; Meger, D.; and Precup, D. 2019. Off-policy deep reinforcement learning without exploration. In *International Conference on Machine Learning*, 2052–2062. PMLR.

Gal, Y.; and Ghahramani, Z. 2016. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, 1050–1059. PMLR.

Gong, D.; Liu, L.; Le, V.; Saha, B.; Mansour, M. R.; Venkatesh, S.; and Hengel, A. v. d. 2019. Memorizing normality to detect anomaly: Memory-augmented deep autoencoder for unsupervised anomaly detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 1705–1714.

Hensman, J.; Matthews, A.; and Ghahramani, Z. 2015. Scalable variational Gaussian process classification. In *Artificial Intelligence and Statistics*, 351–360. PMLR.

Jin, Y.; Yang, Z.; and Wang, Z. 2021. Is Pessimism Provably Efficient for Offline RL? In Meila, M.; and Zhang, T., eds., *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, 5084–5096. PMLR.

Kingma, D. P.; and Welling, M. 2013. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*.

Kostrikov, I.; Nair, A.; and Levine, S. 2022. Offline Reinforcement Learning with Implicit Q-Learning. In *International Conference on Learning Representations*.

Kumar, A.; Agarwal, R.; Ma, T.; Courville, A.; Tucker, G.; and Levine, S. 2021. DR3: Value-Based Deep Reinforcement Learning Requires Explicit Regularization. *arXiv preprint arXiv:2112.04716*.

Kumar, A.; Fu, J.; Tucker, G.; and Levine, S. 2019. Stabilizing off-policy q-learning via bootstrapping error reduction. *arXiv preprint arXiv:1906.00949*.

Kumar, A.; Zhou, A.; Tucker, G.; and Levine, S. 2020. Conservative q-learning for offline reinforcement learning. *arXiv preprint arXiv:2006.04779*.

Kuznetsov, A.; Shvechikov, P.; Grishin, A.; and Vetrov, D. 2020. Controlling overestimation bias with truncated mixture of continuous distributional quantile critics. In *International Conference on Machine Learning*, 5556–5566. PMLR.

Lakshminarayanan, B.; Pritzel, A.; and Blundell, C. 2016. Simple and scalable predictive uncertainty estimation using deep ensembles. *arXiv preprint arXiv:1612.01474*.

Levine, S.; Kumar, A.; Tucker, G.; and Fu, J. 2020. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *arXiv preprint arXiv:2005.01643*.

Liu, H.; and Abbeel, P. 2021. Behavior from the void: Unsupervised active pre-training. *arXiv preprint arXiv:2103.04551*.

Liu, J. Z.; Lin, Z.; Padhy, S.; Tran, D.; Bedrax-Weiss, T.; and Lakshminarayanan, B. 2020a. Simple and principled uncertainty estimation with deterministic deep learning via distance awareness. *arXiv preprint arXiv:2006.10108*.

Liu, Y.; Swaminathan, A.; Agarwal, A.; and Brunskill, E. 2020b. Provably good batch reinforcement learning without great exploration. *arXiv preprint arXiv:2007.08202*.

Mackay, D. J. C. 1992. *Bayesian methods for adaptive models*. Ph.D. thesis, California Institute of Technology.

Miyato, T.; Kataoka, T.; Koyama, M.; and Yoshida, Y. 2018. Spectral normalization for generative adversarial networks. *arXiv preprint arXiv:1802.05957*.

Munos, R. 2003. Error bounds for approximate policy iteration. In *ICML*, volume 3, 560–567.

Neal, R. M. 2012. *Bayesian learning for neural networks*, volume 118. Springer Science & Business Media.

Ovadia, Y.; Fertig, E.; Ren, J.; Nado, Z.; Sculley, D.; Nowozin, S.; Dillon, J. V.; Lakshminarayanan, B.; and Snoek, J. 2019. Can you trust your model's uncertainty? Evaluating predictive uncertainty under dataset shift. *arXiv preprint arXiv:1906.02530*.

Scherrer, B.; Ghavamzadeh, M.; Gabillon, V.; Lesner, B.; and Geist, M. 2015. Approximate modified policy iteration and its application to the game of Tetris. *J. Mach. Learn. Res.*, 16: 1629–1676.

Singh, H.; Misra, N.; Hnizdo, V.; Fedorowicz, A.; and Demchuk, E. 2003. Nearest neighbor estimates of entropy. *American journal of mathematical and management sciences*, 23(3-4): 301–321.

Sutton, R. S.; and Barto, A. G. 2018. *Reinforcement learning: An introduction*. MIT press.

van Amersfoort, J.; Smith, L.; Jesson, A.; Key, O.; and Gal, Y. 2021. On feature collapse and deep kernel learning for single forward pass uncertainty. *arXiv preprint arXiv:2102.11409*.

Van Amersfoort, J.; Smith, L.; Teh, Y. W.; and Gal, Y. 2020. Uncertainty estimation using a single deep deterministic neural network. In *International Conference on Machine Learning*, 9690–9700. PMLR.

Van Oord, A.; Kalchbrenner, N.; and Kavukcuoglu, K. 2016. Pixel recurrent neural networks. In *International Conference on Machine Learning*, 1747–1756. PMLR.

Wu, Y.; Zhai, S.; Srivastava, N.; Susskind, J.; Zhang, J.; Salakhutdinov, R.; and Goh, H. 2021. Uncertainty Weighted Actor-Critic for Offline Reinforcement Learning. *arXiv preprint arXiv:2105.08140*.

Xie, T.; Cheng, C.-A.; Jiang, N.; Mineiro, P.; and Agarwal, A. 2021. Bellman-consistent pessimism for offline reinforcement learning. *Advances in neural information processing systems*, 34: 6683–6694.

Yang, X.; Huang, K.; Goulermas, J. Y.; and Zhang, R. 2017. Joint learning of unsupervised dimensionality reduction and gaussian mixture model. *Neural Processing Letters*, 45(3): 791–806.

Yu, T.; Thomas, G.; Yu, L.; Ermon, S.; Zou, J.; Levine, S.; Finn, C.; and Ma, T. 2020. Mopo: Model-based offline policy optimization. *arXiv preprint arXiv:2005.13239*.

Zhai, S.; Cheng, Y.; Lu, W.; and Zhang, Z. 2016. Deep structured energy based models for anomaly detection. In *International conference on machine learning*, 1100–1109. PMLR.

Zong, B.; Song, Q.; Min, M. R.; Cheng, W.; Lumezanu, C.; Cho, D.; and Chen, H. 2018. Deep autoencoding gaussian mixture model for unsupervised anomaly detection. In *International conference on learning representations*.