A Dataset Statements

- Dataset documentation and intended uses.

  Environments.

  - **IB** IB simulates the characteristics presented in various industrial control tasks, such as wind or gas turbines, chemical reactors, etc. The raw system output for each time step is a 6-dimensional vector including velocity, gain, shift, setpoint, consumption, and fatigue. To enhance the Markov property, the authors stitch the system outputs of the last K timesteps as observations ($K = 30$ by default). The action space is three-dimensional. Each action can be interpreted as three proposed changes to the three observable state variables called current steerings. Original codes can be found at `https://github.com/siemens/industrialbenchmark`.
  - **FinRL** FinRL contains 30 stocks in the pool and the trading histories over the past 10 years. Each stock is represented as a 6-dimensional feature vector, where one dimension is the number of stocks currently owned, another five dimensions are the factor information of that stock. The observation has one dimension of information representing the current account cash balance. The dimension of the action space is 30, corresponding to the transactions of each of the thirty stocks. Original codes can be found at `https://github.com/AI4Finance-LLC/FinRL-Library`.
  - **CityLearn** The CityLearn (CL) environment reshapes the aggregation curve of electricity demand by controlling energy storage in different types of buildings. Domestic hot water (DHW) and solar power demands are modeled in the CL environment. High electricity demand raises the price of electricity and the overall cost of the distribution network. Flattening, smoothing, and narrowing the electricity demand curve help to reduce the operating and capital costs of generation, transmission, and distribution. The observation encodes the states of buildings, including time, outdoor temperature, indoor temperature, humidity, solar radiation, power consumption, charging status of the cooling and heating storage units, etc. The action is to control each building to increase or decrease the amount of energy stored in its own heat storage and cooling equipment. Original codes can be found at `https://github.com/intelligent-environments-lab/CityLearn`.
  - **SalesPromotion** The SalesPromotion environment simulates a real-world sales promotion platform, where the platform operator (a human with some data analysis tools) delivers different discount coupons to each user to promote the sales. The number of discount coupons delivered to the user is from 0 to 5 each day, and the discount will be in the range $[0.6, 0.95]$ when the number of coupons is strictly greater than 0. The coupons have the same discount for a user and the user behavior will be affected by the number of coupons and the discount. A higher discount will promote the sales, but the costs will also increase. The goal for the platform operator is to maximize the total income. Although the output of the policy is continuous, we round the dimension that corresponding to the number of coupons to a integer.

    To build this environment, the user models in the environment are trained from the real-world platform interactive data, which are collected with over 10,000 users from 19/03/2021 to 17/05/2021 (60 days). Each state (the user state) contains the total orders, the average order from the first day, the average fees from the first day, and the day of the week. The user model takes the first three dims of the user state as the input and outputs the user action, which consists of the number of orders and the average fees of a single day.

    We sample 10,000 users to make the offline training dataset, and another 1,000 users to make the offline test dataset. The delivered discount coupons and the user actions are made by the real human operator and real users on the platform. We merge the first 10 days, thus the first day in the offline datasets is 29/03/2021 and the state contains the statistics of the first 10 days. After training the operator's policy, it will be tested in the next 30 days starting from 18/05/2021 with the same users. That is, the horizon of the trajectory is 50 for the training and 30 for the test. This setting follows the real-world scenario (also akin to the backtesting in FinRL). The performance of the behavior policy is calculated over the last 50 days in the training dataset, while the performances of Random and Expert policy are testes in the simulator for the next

30 days. After the 50 days promotion, the users tend to spend more on the platform, thus the Random over the next 30 days perform near to the behavior policy over the 50 days. The Expert policy is made by a senior human operator that delivers the same amount and discount to all the users. Since the users' behavior have changed with the promotion, thus simply imitating the historical promotion actions will fail.

This environment is partly built on our (Polixir Technologies) real-world sales promotion projects. All the offline datasets have gone through the data masking process.

– **Gym-MuJoCo** We set EXCLUDE_CURRENT_POSITIONS_FROM_OBSERVATION to false to include the first dimension of the position in HalfCheetah-v3, Walker2d-v3, and Hopper-v3. We use Gym-MuJoCo: `https://gym.openai.com/envs/#mujoco`.

Dataset description. The datasets consist of trajectories from the industrial benchmark (IB), financial trading (FinRL), city management (CityLearn), sales promotion platform (SP) and Gym-MuJoCo domains. Each item of a dataset consists of (s_t, a_t, r_t, s_{t+1}) tuples. The datasets of each task have been split into training and test dataset.

The name (or identifier) of dataset consists of the domain name (i.e., IB), the data quality (L for low, M for medium and H for high), and the trajectories of the dataset $(100, 1000, 10000)$, except the sales promotion domain. The features of each domain can be found in Table 1. The trajectory length in domains with *Done* may be less than the maximum timesteps.

Table 1: The configuration of environments.

Environment	Observation Shape	Action Shape	Have Done	Max Timesteps
HalfCheetah-v3	18	6	False	1000
Hopper-v3	12	3	True	1000
Walker2d-v3	18	6	True	1000
IB	180	3	False	1000
FinRL	181	30	False	2516
CL	74	14	False	1000
SP	4	2	False	50

Data generation The datasets are collected by Polixir Technologies from Gym-MuJoCo, IB, FinRL and CL environments. Since the actions are continuous in all domains, we use the Gaussian distribution as the policy output. To simulate the real-world data-collection scenarios, we adopt the following steps to collect the data:

1. **Obtain data-collection policies**. For each environment, we use an RL algorithm (SAC) to train on each environment until convergence and record a policy at every epoch. We denote the policy with the highest episodic return during the whole training as the expert policy. Another three levels of policies with around $25\%, 50\%, 75\%$ expert performance are stored to simulate multi-level sub-optimal policies, denoted by low, medium, and high respectively.
2. **Collect data**. With probability 20%, we sample from the trained Gaussian policies to execute, otherwise, we use the mean of Gaussian to execute. We sample from the Gaussian policy during the data collection for two reasons: (1) Because of human manipulation errors, the action demonstrations may be noisy, so we use samples from the policy to reproduce this phenomenon. (2) For training offline RL algorithms: When the transition function is deterministic (e.g., Gym-MuJoCo), the deterministic policy may produce similar and even repetitive trajectories. However, if the transition function is stochastic, we only execute a deterministic policy (mean of the trained Gaussian policy) to generate the datasets.
3. **Make training and test datasets**. For each level, 4 policies with similar returns are selected, among which three policies are randomly selected to collect the training data used for offline RL policy training, and the left one produces the test data for offline validation. The default size of the test data is $1/10$ of the training data for each task. The extra test dataset can be used to design the offline evaluation method for the model selection during training and hyper-parameter selection.

We provide training data with a maximum of 10^4 trajectories and three-level sizes of 10^2, 10^3, and 10^4 trajectories by default, in order to cover both the limited data setting and when data are enough.

Intended uses. Our datasets are provided for offline reinforcement learning with a special focus on real-world applications. The design follows real-world properties like the conservative of behavior policies, limited amounts of data, high-dimensional state and action spaces, and the highly stochastic nature of the environments. The datasets include robotics, industrial control, finance trading, city management and sales promotion tasks with real-world properties, containing three-level sizes of dataset, three-level quality of data to reflect the dataset we will meet in offline RL scenarios. Users can use the dataset to evaluate offline RL algorithms with near real-world application nature.

- URL to website/platform where the dataset/benchmark can be viewed and downloaded by the reviewers.

  The datasets is now accessible at `http://polixir.ai/research/neorl`.

  Besides, repositories describing the dataset and reference library is available at `https://github.com/polixir/NeoRL` and `https://agit.ai/Polixir/NeoRL`.

  You can download the dataset from the web page or follow the instructions of the above two repositories to access the data.

- Author statement that they bear all responsibility in case of violation of rights, etc., and confirmation of the data license.

  We bear all responsibility in case of violation of rights, etc. The datasets are under the Creative Commons Attribution 4.0 License (CC BY). The re-implemented codes of offline RL algorithms and OPE methods are under the Apache 2.0 License.

- Hosting, licensing, and maintenance plan.

  The datasets are hosted by Polixir Technologies and are publicly available at `http://polixir.ai/research/neorl`. Polixir Technologies will provide long-term maintenance. The current license for the datasets is CC BY.

- Links to access the dataset and its metadata.

  The datasets is now accessible at `http://polixir.ai/research/neorl`. The following link is for the metadata on the `paperswithcode`. `https://paperswithcode.com/dataset/neorl`

- Long-term preservation: It must be clear that the dataset will be available for a long time, either by uploading to a data repository or by explaining how the authors themselves will ensure this.

  Since the volume of current datasets are too large, e.g., the file size of each of the FinRL 1000 tasks can be more than 3GB, we store them on our company data center and provide a persistent public link to access it. We are considering moving it to a data repository.

- Explicit license: Authors must choose a license, ideally a CC license for datasets, or an open source license for code (e.g. RL environments).

  The datasets are under the Creative Commons Attribution 4.0 License (CC BY). The re-implemented codes of offline RL algorithms and OPE methods are under the Apache 2.0 License.

- Highly recommended: a persistent dereferenceable identifier (e.g. a DOI minted by a data repository or a prefix on identifiers.org) for datasets, or a code repository (e.g. GitHub, GitLab,...) for code. If this is not possible or useful, please explain why.

  The codes for NeoRL are available at `https://github.com/polixir/NeoRL` or `https://agit.ai/Polixir/NeoRL`, including the loading of the data, our re-implemented offline RL algorithms and OPE methods. The datasets are hosting our our company's server, so they do not conatin an identifier currently.

- For benchmarks, the supplementary materials must ensure that all results are easily reproducible.

  We list the implementation details in the Appendix C, F and G.