Ensembling Diverse Policies Improves Generalizability of Reinforcement Learning Algorithms in Continuous Control Tasks

Abilmansur Zhumabekov
University of Alberta
Edmonton, Canada
zhumabek@ualberta.ca

Daniel May
University of Alberta
Edmonton, Canada
dcmay@ualberta.ca

Tianyu Zhang
University of Alberta
Edmonton, Canada
tzhang6@ualberta.ca

Aakash Krishna GS
University of Alberta
Edmonton, Canada
krishnag@ualberta.ca

Omid Ardakanian
University of Alberta
Edmonton, Canada
ardakanian@ualberta.ca

Matthew E. Taylor
University of Alberta & Alberta Machine Intelligence Institute (Amii)
Edmonton, Canada
matthew.e.taylor@ualberta.ca

ABSTRACT

Deep Reinforcement Learning (DRL) algorithms have shown great success in solving continuous control tasks. However, they often struggle to generalize to changes in the environment. Although retraining may help policies adapt to changes, it may be quite costly in some environments. Ensemble methods, which are widely used in machine learning to boost generalization, have not been commonly adopted in DRL for continuous control applications. In this work, we introduce a simple ensembling technique for DRL policies with continuous action spaces. It aggregates actions by performing weighted averaging based on the uncertainty levels of the policies. We investigate its zero-shot generalization properties in a complex continuous control domain: the optimal control of home batteries in the CityLearn environment — the subject of a 2022 international AI competition. Our results indicate that the proposed ensemble has better generalization capacity than a single policy. Further, we show that promoting diversity among policies during training can reliably improve the zero-shot performance of the ensemble in the test phase. Finally, we examine the merits of the uncertainty-based weighted averaging in an ensemble by comparing it to two alternative approaches: unweighted averaging and selecting the action of the least uncertain policy.

KEYWORDS

Reinforcement Learning, Generalization, Continuous Control, Energy

1 INTRODUCTION

Deep reinforcement learning (DRL) algorithms have attained remarkable performance in a variety of challenging continuous control tasks such as locomotion and manipulation [6, 8, 11]. However, DRL agents have been shown to have limited generalization capabilities, tending to be overly specialized to their environment and failing to perform optimally when faced with perturbations [15, 36]. This is especially relevant for DRL agents trained in a simulator or digital twin for deployment in a real-world setting. The differences between the training and deployment (test) environment include state space, transition dynamics, observation function, etc. [15].

Proc. of the Adaptive and Learning Agents Workshop (ALA 2023), Cruz, Hayes, Wang, Yates (eds.), May 9-10, 2023, Online, https://alaworkshop2023.github.io/. 2023.

Closing this generalization gap is the focus of a broad body of research. For example, recent works have shown that generalization techniques from supervised learning, such as L2 regularization, dropout, data augmentation, and batch normalization, prove useful in DRL as well [5, 20]. Another generally accepted approach to boosting the generalization properties of machine learning (ML) models is to build ensembles of diverse models [9, 12]. Despite the prevalence of ensembling in the context of general ML, there remains a scarcity of research exploring the use of (diverse) ensemble methods for continuous control tasks in DRL. In particular, their use for improving generalization to perturbations in the environment has been limited to date.

In this study, we introduce the 'Diverse σ-weighted ensemble' for continuous action spaces in DRL (see Section 3), and examine its zero-shot generalization properties on the data and task of the 2022 CityLearn Challenge [16] — household battery control for demand response, which is a challenging, partially observable continuous control task. Our key contribution is training *diverse* DRL policies and combining them according to their uncertainty in the given task. The main insights of this work can be summarized as follows:

(1) Compared to using only a single policy, the proposed ensembling method performs significantly better in the test phase and resists overfitting for much longer;
(2) Promoting policy diversity in ensembles can significantly improve their zero-shot test performance, albeit the extent of improvement varies across different ensembling approaches;
(3) The effectiveness of the proposed ensembling method comes from its ability to leverage diversity not only in the actions but also in the uncertainty levels of its members.

2 BACKGROUND

We introduce the demand response problem in an electrical grid comprised of renewable and distributed energy resources, followed by the definition of the battery control task in the CityLearn environment. Furthermore, we briefly explain the DRL algorithm used in this work and how the ensembles of policies may be used in RL. Finally, we give an overview of policy diversity in RL.

2.1 Demand Response for Electric Grids

The adoption of distributed energy resources (DERs), such as solar panels and electric energy storage systems, can offset, shift, or reduce electricity and emission costs for the entire grid and individual customers. However, the intermittent nature of DER usage and generation patterns poses a significant challenge to the stability of the traditional grid [13]. One prominent approach to tackling this challenge is to employ *demand response* (DR). The US Department of Energy defines DR as "... changes in electric usage by end-use customers from their normal consumption patterns in response to changes in the price of electricity over time, or to incentive payments ..." [7]. DR approaches are broadly classified into direct DR (direct, external control of end-user's assets) or price-based DR, which uses real-time fluctuation of a monetary incentive signal to nudge end-user behavior.

Intelligent algorithms are needed to perform DR effectively. Given the success of RL in other continuous control tasks, a body of research investigating the application of RL to DR has started to develop [35]. Here, we focus on price-based DR, in which homeowners aim to optimize their energy use and battery operation based on a given price signal which is an exogenous variable.

2.2 The Battery Control Task in CityLearn

To facilitate research in applying RL to DR, Vazquez et al. published the CityLearn environment [34] on the basis of OpenAI Gym [3]. Within CityLearn, we focus on the price-based DR task defined in the 2022 CityLearn Challenge [16]: controlling charging and discharging of a household battery, given the time-series input about the building's energy demand and solar generation, electricity pricing, carbon emission rate, as well as various weather signals (details provided in Section 2.3).

While CityLearn supports both building-level (single-agent) and district-level (multi-agent) objectives, we focus on the single-agent metrics in this work. Hence, the objective of each house is twofold: to minimize the electricity cost, as well as the carbon emission cost. Notice that minimizing the electricity cost is not equivalent to minimizing emissions, because electricity prices in today's electricity markets do not solely reflect the carbon intensity of power plants. The costs are defined as follows:

$$C_{price} = \sum_{t=1}^{T} C_{price}(t) = \sum_{t=1}^{T} p_t * (d_t - g_t + b_t)^+$$

$$C_{emission} = \sum_{t=1}^{T} C_{emission}(t) = \sum_{t=1}^{T} c_t * (d_t - g_t + b_t)^+$$

Where $C_{price}(t)$ is the electricity cost and $C_{emission}(t)$ is the emission cost. t is the time-step with the duration of 1 hour and T is the duration of the control task in hours. p_t and c_t are respectively the electricity pricing and carbon emission rates per unit of net energy demand (the expression inside brackets). d_t is the non-shiftable electricity demand of a household, g_t is the energy generated by its solar panels, and b_t is the amount of energy charged into the battery (negative values imply discharging). The + superscript indicates that negative values are clipped to 0.

We adopt the normalized scoring employed in the 2022 CityLearn challenge[16]:

$$\hat{C}_{price} = \frac{C_{price}}{C_{price}^{noop}},$$

$$\hat{C}_{emission} = \frac{C_{emission}}{C_{emission}^{noop}},$$

$$\hat{C} = \frac{1}{2}(\hat{C}_{price} + \hat{C}_{emission}), \tag{1}$$

where C_{price}^{noop} and $C_{emission}^{noop}$ are respectively C_{price} and $C_{emission}$ with b_t set to 0, i.e., costs with no-battery or no control.

2.3 Reinforcement Learning

To apply RL techniques to the battery control problem, the task can be formulated as a partially observable Markov decision process (POMDP), which is a tuple $\langle S, \Omega, O, A, T, R \rangle$ [32].

S is the set of states s, which are not directly accessible but rather have to be inferred from the observations o coming from the continuous set of observations Ω. Observations are generated by the probability density function $O : S \times A \times \Omega \rightarrow [0, \infty)$, called the observation function. A is the continuous set of actions. The transition function $T : S \times A \times S \rightarrow [0, \infty)$ represents the probability density of the next state $s_{t+1} \in S$ given the current state $s_t \in S$ and action $a_t \in A$. Finally, $R : S \times A \times S \rightarrow \mathbb{R}$ is the reward function.

An RL agent aims to maximize the expected return, the discounted sum of future rewards, by learning a policy π [33]. The policy π is a probabilistic mapping of observations $o \in \Omega$ to actions $a \in A$. Therefore the RL objective can be formulated as follows:

$$J(\pi) = \mathbb{E}_{\pi}[\sum_{t=0}^{H} \gamma^t r_t(o_t, a_t)] \tag{2}$$

Where r_t is the reward for taking action a_t when observing o_t, $\gamma \in (0; 1]$ is the discount factor, and H is the duration of an episode.

If the RL agent uses a neural network to map observations to actions, it is called a deep reinforcement learning (DRL) agent. There exists a variety of algorithms to train DRL agents [2]. In this work, we employ Soft Actor-Critic (SAC) [11], an established DRL algorithm known for its relative robustness and sample efficiency.

In the CityLearn environment, observation $o \in \Omega$ is a vector with information about the system in the past hour: month, day of the month, the hour of the day, household electricity demand, solar generation, battery state of charge (SoC), net demand (electricity demand - solar generation + charging), and weather. Weather information consists of outdoor temperature, humidity, diffuse and direct solar irradiance, as well as their forecasted values for 6, 12, and 24 hours ahead. Detailed information on all observation features is given in Appendix B. We note that this is a partially observable task as o does not fully describe s. s includes a perfect, infinite horizon forecast of several, multi-variate time-series such as non-shiftable load, solar generation, and carbon intensity.

Provided with an observation o, the DRL agent must choose an action $a \in [-1; 1]$ that determines the battery (dis)charging rate in the upcoming hour and that maximizes the RL objective in Equation 2. Rewards are carefully designed in Section 4.2 so that maximizing

the RL objective corresponds to minimizing the costs that we care about (Equation 1).

2.4 Ensembles in Deep Reinforcement Learning

Ensembles are an established tool to boost the generalization capabilities of ML models [9, 29]. However, their usage for improving generalization in RL is underexplored. In this subsection, we summarize existing works investigating ensembling techniques for RL algorithms and highlight differences in our approach.

An et al. [1] use an ensemble of Q networks in an offline RL setting. They estimate the Q value of a state-action pair by choosing the minimal value outputted by the set of Q networks, which leads to the penalization of out-of-distribution actions for which there is high uncertainty in Q-value estimates. Ensembling both critics and actors proved useful in stabilizing learning and improving exploration during training, according to Lee et al. [17], where the mean and standard deviation of Q-value estimates are used to reweight Bellman backups and to perform UCB exploration. Unlike [1] and [17], our focus is on the case when the agent is allowed to learn online during training (i.e., influence the environment and receive feedback), and it is challenged to zero-shot generalize to a test environment that will be different from the training environment.

Yang et al. [37] use three different DRL algorithms in an ensemble: PPO[31], A2C[23], and DDPG [19] to trade stock shares. In each quarter, only one of the algorithms is used to trade, but all three can be evaluated in the background. The algorithm with the best evaluation score is selected to trade in the next quarter. According to the authors, different models are sensitive to different trends, so ensembles should work better than any of their members alone. Although their experiments demonstrate the effectiveness of the strategy, it has a limiting assumption that evaluation scores can be computed for the algorithms that did not participate in trading. In contrast, our 'Diverse σ-weighted' ensembling approach does not have the requirement of evaluating all policies and picking one of them. Instead, it simply combines all of their actions with weighted averaging.

Ghosh et al. [10] show that ensembles can improve the generalization performance of RL agents. Their method of combining actions is shown to work for discrete action spaces, but transferring it to continuous action spaces is a non-trivial task. In contrast, our work is focused on continuous action spaces.

2.5 Policy Diversity

Established ensembling methods in ML are highly effective largely because they leverage some form of diversity, which may come from an auxiliary penalty term imposed on outputs or from variations in training data, input representations, learning algorithms, etc. [28, 29, 39]. For this reason, one of the goals of this paper is to investigate the effect of policy diversity on the DRL ensemble's generalization capacity.

In RL, diversity can stem from variations in the environment or the agent behaviors (policies) [22]. In this paper, we focus on policy diversity, which can be quantified by measuring the difference between trajectories (state-action or observation-action sequences) traversed by the policies [18, 21] or by evaluating the disparity in policy actions when provided with the same states/observations [22, 27, 38].

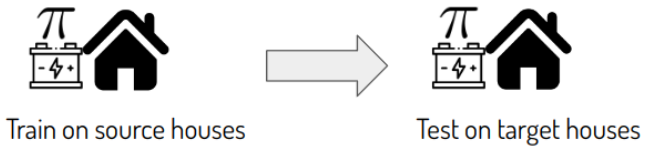


Figure 1: Single Policy Training and Evaluation Process.

In our study, we employ the Diversity via Determinants (DvD) method proposed by Parker-Holder et al. [27]. It adds an auxiliary diversity term to the objective, which encourages policies to output diverse actions when provided with the same observations:

$$J(\phi_1, \phi_2, ..., \phi_N) = \sum_{i=1}^{N} \mathbb{E}_{\tau \sim \pi_{\phi_i}}[R(\tau)] + \lambda Div(\phi_1, \phi_2, ..., \phi_N), \quad (3)$$

where $\phi_1, ..., \phi_N$ are parameters of N policies, τ is the trajectory traversed by a policy in an episode, and R is the return (discounted sum of rewards). Importantly, the diversity term on the right captures the volume spanned by policies in a behavioral manifold. In other words, it measures the degree to which outputs of different policies are different from each other when faced with the same observations. For more details, we refer the reader to [27].

One of the benefits of DvD is that it is task-agnostic, meaning it does not require hand-crafting policy representations for a specific domain. Moreover, it allows tuning the degree of diversity by controlling λ — the importance coefficient of the diversity objective. Last but not least, it is easy to implement thanks to a reference implementation [30].

3 DIVERSE σ-WEIGHTED ENSEMBLING TECHNIQUE

When training a DRL agent on a given environment, only one policy π_ϕ is typically learned. With SAC, as with most actor-critic models, the actor's policy is defined as $\pi_\phi = \langle \mu_\phi, \sigma_\phi \rangle$, meaning the actor is modeling a Normal distribution $\mathcal{N}$ with characterizing parameters mean μ_ϕ and standard deviation σ_ϕ. During training, the agent samples this distribution stochastically so that

$$\hat{a}_{train}(o_t) \sim \mathcal{N}(\mu_\phi(o_t), \sigma_\phi(o_t))$$

while during evaluation (test), actions are deterministically selected:

$$\hat{a}_{eval}(o_t) = \mu_\phi(o_t)$$

In order to constrain actions, SAC further applies tanh function as well as scaling [11]:

$$a_{train} = c_a \tanh(\hat{a}_{train})$$

$$a_{eval} = c_a \tanh(\hat{a}_{eval})$$

The action scaling coefficient c_a is a hyperparameter (for details on all hyperparameters, see Appendix C). Together with the scaling coefficient, tanh function puts the actions into $[-c_a; c_a]$ range. We refer to this procedure as the 'Single Policy' approach and depict its training and evaluation pipeline in Figure 1.

To improve zero-shot generalization on a perturbed environment, we propose an ensembling method for continuous action spaces.

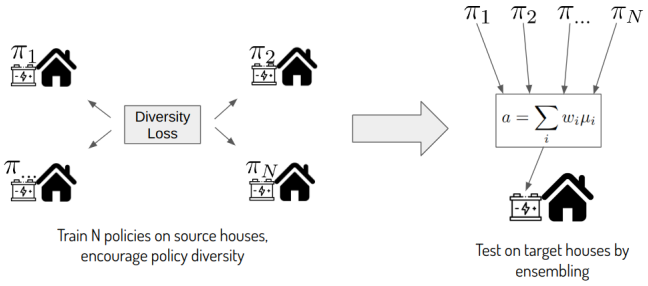


Figure 2: Ensemble Policy Training and Evaluation Process.

We train multiple actors (with one shared critic) in separate (but identical) environments and aggregate them in an ensemble during the test phase. The ensemble's output is a weighted average of its individual members' actions, where each weight is inversely proportional to the degree of uncertainty of the policy. We use $\sigma_\phi(o)$ as a proxy for this uncertainty.

More concretely, we train N policies that could be represented as follows:

$$\pi_{\phi_i} = \langle \mu_{\phi_i}, \sigma_{\phi_i} \rangle, \text{ for } i = 1, 2, ..., N$$

During training, each policy's action is sampled stochastically and executed in a separate training environment:

$$\hat{a}^i_{train}(o_t) \sim \mathcal{N}(\mu_{\phi_i}(o_t), \sigma_{\phi_i}(o_t)),$$
$$a^i_{train} = c_a \tanh(\hat{a}^i_{train})$$

All N actors are trained in parallel, and their loss is augmented with the diversity term [27] discussed in Section 2.5.

During evaluation (test), we combine the outputs of these policies into one action that is executed in the test environment:

$$a^\sigma_{eval}(o_t) = \frac{\sum_{i=1}^N w_i a^i_{eval}(o_t)}{\sum_{i=1}^N w_i}, \tag{4}$$

where $a^i_{eval}(o_t) = c_a \tanh(\mu_{\phi_i}(o_t))$ and $w_i = \frac{1}{\sigma_{\phi_i}(o_t)}$. This approach is illustrated in Figure 2, and further referred to as '(Diverse) σ-weighted ensemble'.

The motivation behind this weighting of actions is that standard deviations $\sigma_{\phi_i}(o_t)$ measure the uncertainties of their corresponding policies π_{ϕ_i}. Distinct policies go through different experiences and updates during training, so they might have varying degrees of certainty in their actions when faced with a new observation o_t in the test environment. This disparity can increase further when policy diversity is promoted during training. Thus, by using standard deviations, we are taking into account the confidence levels of different policies, which, as experiments reveal, leads to better performance at test time.

To confirm that policies in the 'Diverse' ensemble generate more diverse actions, compared to the 'Non-diverse' ensemble, we calculate their standard deviation:

$$\mathcal{D}^a(o_t) = \sqrt{\frac{\sum_{i=1}^N (a^i_{eval}(o_t) - \bar{a}_{eval}(o_t))^2}{N}}, \tag{5}$$

where $\bar{a}_{eval}(o_t)$ is the mean of the actions chosen by the policies in an ensemble:

$$\bar{a}_{eval}(o_t) = \frac{1}{N} \sum_{i=1}^N a^i_{eval}(o_t). \tag{6}$$

Furthermore, policy diversity can also be manifested in the diversity of uncertainties among policies. To measure it, we calculate the coefficient of variation of σ_{ϕ_i} values:

$$\mathcal{D}^\sigma(o_t) = \frac{1}{\bar{\sigma}(o_t)} \sqrt{\frac{\sum_{i=1}^N (\sigma_{\phi_i}(o_t) - \bar{\sigma}(o_t))^2}{N}}, \tag{7}$$

where $\bar{\sigma}(o_t) = \frac{1}{N} \sum_{i=1}^N \sigma_{\phi_i}(o_t)$ — is the average of uncertainties. We use this metric because it is not affected by the scale of σ_{ϕ_i} values. We cannot use the coefficient of variation for measuring action diversity because a^i_{eval} can be negative, but using the standard deviation metric is acceptable since the action values are bounded: $a \in \{-c_a; c_a\}$.

In the Results and Discussion (Section 5) we report values $\mathcal{D}^a$ and $\mathcal{D}^\sigma$ that are averages of $\mathcal{D}^a(o_t)$ and $\mathcal{D}^\sigma(o_t)$ across all target (test) buildings and over the entire test episode (see Section 4.1). We choose these metrics because they are easy to interpret and implement, and they give insights into the benefits of our proposed ensembling method (see Section 5.2).

4 METHODOLOGY AND EXPERIMENTAL PROCEDURE

In this section, we describe our experimental setup by providing details on the dataset and its train-test split in Section 4.1. We then explain the reward design and training procedure in Section 4.2.

4.1 Dataset and Cross-Validation

We use the dataset from CityLearn 2022 challenge [16], which contains 1-hour resolution data for a period of 1 year obtained from a neighborhood of 17 single-family houses in Fontana, California [26]. After examining the hourly power consumption profiles of each building [25] and discussing them with the dataset's publishers, we decided to omit 2 buildings (numbered 12 and 15) with highly abnormal consumption profiles. These abnormalities could have resulted from malfunctioning measurement equipment. Next, to perform cross-validation, the remaining 15 buildings were partitioned into 3 groups of 5 buildings each: the first group (buildings 1 through 5), the second group (buildings 6 through 10), and the third group (buildings 11, 13, 14, 16, 17).

In all experiments, to attain statistically significant results, we perform 3-fold cross-validation with 5 independent trials in each. For every fold, we train an algorithm on one group (5 source buildings) and test on the remaining two groups (10 target buildings). We perform statistical comparisons using the Mann-Whitney U test (also called the Wilcoxon rank-sum test) [24].

We further adopt the most difficult deployment setting from Nweye et al. [25], restricting training to the first 5 months of data and performing testing on the remaining 7 months. This setup mimics a to-scale deployment scenario from an accurately simulated training environment with 'few' data streams to a real environment with many data streams.

For each experiment, we report the zero-shot performance on the 7 months of the target building data in terms of metrics established in Section 2.2, averaged across all folds and trials (15 samples).

4.2 Training Procedure and Reward Design

First, it is important to clarify how training episodes are counted. For a single policy, one training episode is equivalent to one pass through the first 5-month of data for 5 source buildings. For ensembles of size N (e.g., N=4), when each ensemble member goes through the same data once, we count it as N training episodes completed by the ensemble. While counting training episodes may seem involved, one test episode simply corresponds to one pass through the 7-month data for 10 target buildings.

On a related note, our SAC algorithm performs random exploration at the beginning of training, when it samples actions randomly from a uniform distribution and saves resulting transitions to the replay buffer. The duration of that period must be standardized for ensembles of different sizes. In our experiments (including Appendix A), we consider ensembles of sizes N=1, 2, 4, 8 — where N=1 corresponds to the single policy. When the biggest ensemble of size N=8 goes through the training data once — its members gather 8 episodes of cumulative random experience. To ensure a fair comparison, each ensemble must collect the same amount of cumulative random exploration experience. This is achieved by fixing the number of exploration episodes at 8.

Finally, we describe our reward function that encourages minimization of the cost in Equation 1. It consists of price and emission components:

$$r_t^{price} = C_{price}^{noop}(t) - C_{price}(t),$$
$$r_t^{emission} = C_{emission}^{noop}(t) - C_{emission}(t)$$

At the end of the random exploration period, we calculate the means and standard deviations of observations and rewards (separately for each component) and use these to normalize them. Normalization is finalized by scaling the reward up by a factor of c_r (see Appendix C).

Normalized reward terms $\hat{r}_t^{price}$ and $\hat{r}_t^{emission}$ are then combined into the building reward:

$$r_t = \frac{1}{2}(\hat{r}_t^{price} + \hat{r}_t^{emission})$$

5 RESULTS AND DISCUSSION

We now present and discuss the results of our experiments. We start with Section 5.1, which examines the generalization capability of the σ-weighted ensembling method (See Section 3) in both diverse and non-diverse settings, comparing it to the canonical 'Single Policy' approach (Figure 1). For the sake of comparison, we also use two rule-based controllers as baselines:

- RBC_{ToU} - The Time-of-Use Peak Reduction strategy that has been deployed in real life on the majority of houses from the dataset [25]. It charges the battery from 9 am to 12 pm and discharges from 6 pm to 9 am. Both charging and discharging rates are 2kW/h (31.25% of battery capacity). Discharging is only allowed when the battery is at least 25% full.
- RBC_{HC} - The Hand-Crafted controller of our design. We used its slightly modified version as a part of our solution when participating in the CityLearn 2022 challenge [16]. Its implementation details are given in Appendix D.

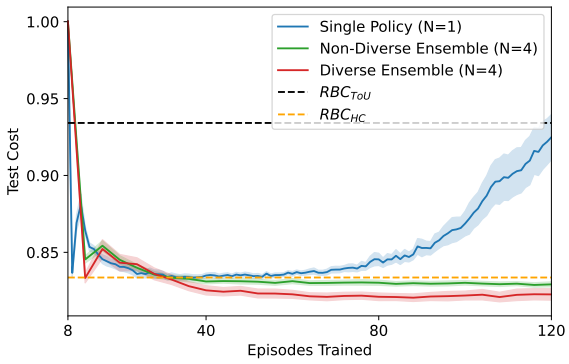


Figure 3: Zero-shot costs averaged across target buildings, comparing Single Policy vs. Diverse and Non-diverse σ-weighted ensembles. All agents train on the first 5 months of the source buildings data and are tested on the remaining 7 months of the target buildings data. The ensembles achieve lower test costs compared to the Single Policy and resist overfitting for longer. The Diverse Ensemble outperforms its Non-Diverse counterpart.

We then study the benefits of performing σ-weighted averaging of policy actions in Section 5.2 by comparing it with two alternative action selection mechanisms:

- Simple-averaging — combining actions using the unweighted average from Equation 6.
- Min-σ — selecting only one action $\mu_{\phi_i}(o_t)$ with the smallest $\sigma_{\phi_i}(o_t)$ and ignoring the rest.

All experiments are conducted with an ensemble size of $N = 4$. Refer to Appendix A for more details on the choice of N.

5.1 Diverse Ensembles of DRL policies

In this experiment, using the Single Policy approach (Figure 1) as a baseline, we examine zero-shot generalization capabilities of the 'Diverse σ-weighted' ensemble proposed in Section 3. We also compare it to its non-diverse ablation, labeled 'Non-diverse σ-weighted' ensemble, to study the role of policy diversity.

Figure 3 shows zero-shot costs (lower is better) on target buildings plotted against the number of training episodes completed on source buildings for each approach. The shaded areas span standard error over 15 trials from the validation procedure described in Section 4.1, while the lines denote the averages. Since we evaluate test scores after every pass through the 5-month training data, ensembles of size N have values only for every Nth training episode completed (see Section 4.2).

Table 1 shows average zero-shot costs across folds for target buildings after the 40th, 80th, and 120th episodes of training on source buildings. We notice that the test costs of all methods are unstable at the initial stage of training. They are remarkably low after the first N training episodes that follow 8 random-exploration episodes, so we include the test costs obtained after the 8+Nth training episode as well. We mark in bold the results for which

Table 1: For the σ-weighted ensembles and the Single Policy: zero-shot test costs (in %) on target buildings obtained after 8+Nth, 40th, 80th, 120th episodes of training on source buildings. In bold: ensemble outperforms Single Policy with $p \leq 0.05$ within any column; * — diverse ensemble outperforms the other methods with $p \leq 0.05$ within any column.

Method	8+N	40	80	120
Single Policy (N=1)	83.66	83.49	84.54	92.49
Non-diverse ensemble (N=4)	84.54	**83.11**	**83.02**	**82.92**
Diverse ensemble (N=4)	83.33	**82.53***	**82.11***	**82.26***

ensembles outperform the Single Policy approach with $p \leq 0.05$ when comparing with Single Policy's every column. We also denote with * the cases when one method outperforms the others in every column (e.g., the diverse ensemble evaluated after 40 episodes outperforms other methods evaluated after 8+N, 40, 80, and 120 episodes).

We note that the Non-diverse σ-weighted ensemble converges to lower cost values compared to the Single Policy and resists overfitting to training data for much longer. Further, the diverse ensemble outperforms its non-diverse counterpart regardless of the duration of the training with a statistical significance of $p \leq 0.05$, demonstrating that policy diversity further improves the zero-shot generalization ability of the σ-weighted ensemble.

To support the claims above, in Figure 4, we plot the training costs achieved throughout the training process. During the initial random exploration phase (see Section 4.2 for details), the training costs are very high, so we omit them in the plot for a better comparison. We note that, as expected, the training costs decrease monotonically for all methods. This stands in contrast to the Single Policy's test cost, which noticeably diverges after 40 episodes, while both σ-weighted ensembles maintain low test cost values even after 120 episodes of training (Figure 3). These observations confirm that the ensembles exhibit higher resistance against overfitting to training data compared to the Single Policy approach.

With respect to training costs, the Diverse and Non-diverse ensembles perform equally, and both do worse than the Single Policy approach (Figure 4). Comparing that to Figure 3 further affirms that the differences in test costs do not come from the differences in training costs.

To confirm that policies in the diverse ensemble indeed output more diverse actions, we plot the diversity metric for actions $\mathcal{D}^a$ (Equation 5) in Figure 5a for both 'Diverse' and 'Non-diverse' σ-weighted ensembles of size N=4. From this plotting, it can be seen that policies in the diverse ensemble differ in their decisions much more than policies in the non-diverse ensemble. Similarly, in Figure 5b, we illustrate the diversity in uncertainty levels $\mathcal{D}^\sigma$ (Equation 7) in diverse and non-diverse ensembles. We notice that policies in the diverse ensemble have greater variation in their uncertainties as well.

5.2 Comparison of Ensembling Methods

In this section, we investigate the effect of the ensembling method choice and its role in leveraging policy diversity. To do so, we compare the σ-weighted ensembling technique to two baselines introduced at the beginning of Section 5, the Simple-averaging ensemble and the Min-σ ensemble, in the diverse and non-diverse setting.

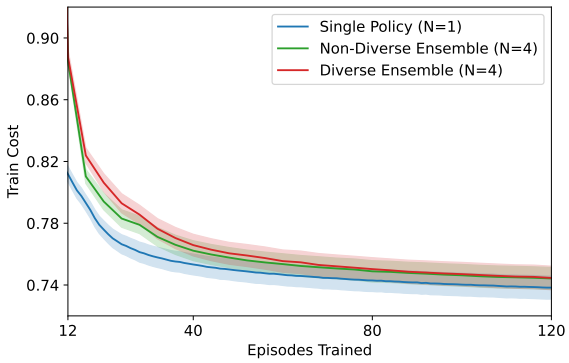
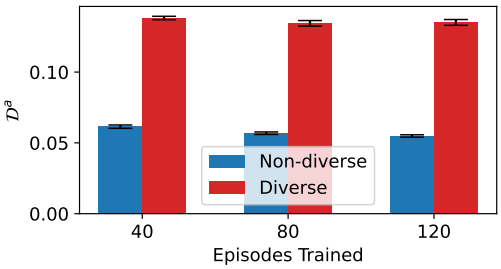
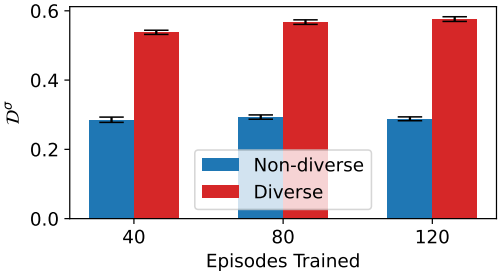


Figure 4: Training costs averaged across source buildings, comparing Single Policy vs. Diverse and Non-diverse σ-weighted ensembles. All agents train on the first 5 months of source building data and, for this plot, are evaluated on the same data. The ensembles do not outperform the Single Policy on training data, so the differences in test performances do not result from the differences in training performances.



(a) Diversity in Policy Actions



(b) Diversity in Policy Uncertainties

Figure 5: Comparison for (a) diversity in actions a^i_{eval} and (b) diversity in uncertainties σ_{ϕ_i} for policies in Non-diverse and Diverse σ-weighted ensembles. The error bars denote the standard errors. The 'Diverse' ensemble exhibits higher diversity in both actions and uncertainties.

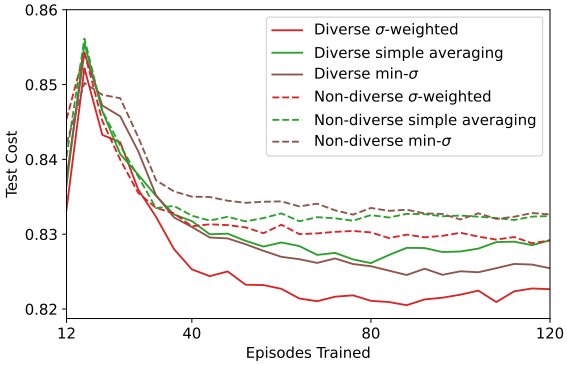


Figure 6: σ-weighted averaging vs. Simple-averaging vs. Min-σ action selection. The plot suggests that the σ-weighted ensemble achieves the lowest test costs in both 'Diverse' and 'Non-diverse' settings, and that diversity is helpful to all ensembling methods.

Table 2: Zero-shot test costs of different ensembling methods when trained with and without diversity, obtained after 8+Nth, 40th, 80th, and 120th episodes of training. In bold: diverse method outperforms its non-diverse version with $p \leq 0.05$ within the same column.

Method	8 + N	40	80	120
Non-diverse Simple-averaging	84.03	83.25	83.26	83.24
Non-diverse Min-σ	84.21	83.50	83.35	83.26
Non-diverse σ-weighted	84.54	83.11	83.02	82.92
Diverse Simple-averaging	83.65	83.18	**82.61**	**82.92**
Diverse Min-σ	83.75	**83.10**	**82.57**	**82.54**
Diverse σ-weighted	**83.33**	**82.53**	**82.11**	**82.26**

Figure 6 shows the test costs of these approaches averaged over 15 trials. We do not shade the standard errors to avoid clutter. To focus on the differences between each approach, we skip the test cost evaluated after 8 random exploration episodes (where all methods get a cost of about 1). The plot suggests that all methods benefit from enhanced policy diversity and that σ-weighted ensembles achieve lower zero-shot test costs compared to the alternatives in both diverse and non-diverse training scenarios.

Table 2 compares zero shot costs of the tested ensembling methods. We boldface the cases where a diverse ensemble outperforms its non-diverse version with $p \leq 0.05$. From both the table and Figure 6, it is clear that Min-σ and σ-weighted ensembles are better at leveraging diversity than the Simple-averaging method.

Further statistical analysis shows that the diverse σ-weighted ensemble significantly outperforms ($p \leq 0.05$) the diverse Simple-averaging method when tested after 40, 80, and 120 training episodes. The only difference between these approaches is that σ-weighted averaging leverages the diversity in uncertainty levels σ_{ϕ_i} among ensemble members π_{ϕ_i}, while Simple-averaging does not. Therefore, it is reasonable to conclude that leveraging the diversity of uncertainties in an ensemble improves the generalization performance.

Moreover, Table 2 suggests that diverse Simple-averaging successfully outperforms its non-diverse counterpart when tested after long training but not so after shorter periods of training. It seems that exploiting the diversity in actions alone, without accounting for uncertainties, has a positive but limited effect on generalization. In contrast, the diverse σ-weighted method, which leverages diversity in both actions and uncertainties, outperforms its non-diverse counterpart more consistently. These results indicate that leveraging the variations in both actions and uncertainties (Figure 5) is important and that diverse σ-weighted averaging gains boosts in zero-shot test performance from both.

Next, statistical comparison of diverse σ-weighted and diverse Min-σ approaches does not report a significant difference in their performance. However, we note that these results are given for the best λ (importance coefficient of the DvD diversity term, as described in Section 2.5) for each ensemble type, found from the search space $\lambda \in \{0.2, 0.4, 0.6, 0.8\}$. Details on λ values for each ensemble are given in Appendix C. Figure 7 compares test costs of Min-σ and σ-weighted ensembles under different values of λ. From the plots, it is evident that the σ-weighted ensemble is more robust to changes in the λ hyperparameter. This outcome suggests the importance of considering the outputs of all policies, not just the most confident one.

To sum up, this subsection shows that the σ-weighted ensemble reliably outperforms the alternatives by leveraging the diversity in both actions and uncertainties of all of its members. Crucially, the disparity found in zero-shot generalization properties of these few ensembling approaches prompts further research into a more extensive set of ensembling techniques.

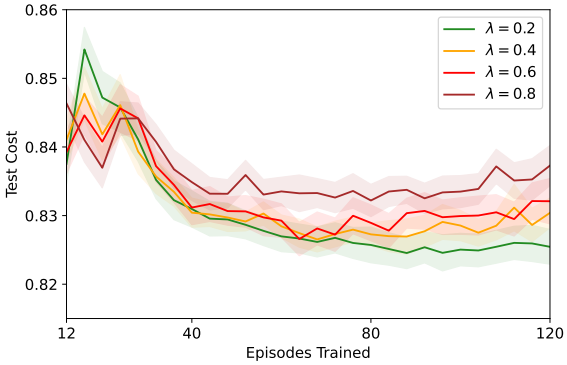
6 CONCLUSION

In this work, we proposed the 'Diverse σ-weighted ensemble' of DRL policies for continuous control tasks, which weighs the actions of its members based on their degrees of uncertainty. We then performed experiments on a realistic battery control task in CityLearn. First, we showed that the proposed ensemble can improve zero-shot generalization to environmental changes in continuous control tasks. Next, we demonstrated that promoting policy diversity in ensembles significantly and reliably improves test performance further. Lastly, we found that the effectiveness of the Diverse σ-weighted ensemble stems from its ability to leverage diversity in both actions and uncertainties of all of its members.

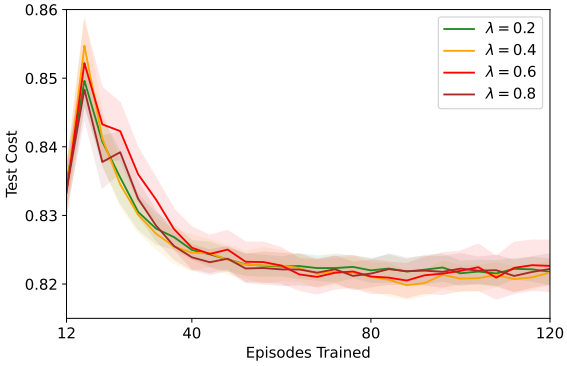
Future work will focus on extending our experiments to other continuous control benchmarks with various types of environmental changes. In addition, it is important to compare the σ-weighted ensembling method with a bigger set of ensembling techniques and deeper explore the role of DRL algorithm choice, critic centralization, and ensemble member diversity.

ACKNOWLEDGMENTS

We would like to thank Dr. Zoltan Nagy for the insightful discussions about CityLearn and battery control, and for his efforts in making the dataset from CityLearn 2022 challenge public. We also want to acknowledge Zhihu Yang for his great contribution



(a) Min-σ ensemble



(b) σ-weighted ensemble

Figure 7: Comparison of Min-σ and σ-weighted ensembles under different diversity importance coefficients λ. Shaded regions denote the standard error. The plots suggest that the σ-weighted ensembling method is more robust to the choice of the λ hyperparameter.

in developing the SAC algorithm used in our work. Part of this work has taken place in the Intelligent Robot Learning (IRL) Lab at the University of Alberta, which is supported in part by research grants from the Alberta Machine Intelligence Institute (Amii); a Canada CIFAR AI Chair, Amii; Compute Canada; Huawei; Mitacs; and NSERC.

APPENDIX

We refer the reader to https://tinyurl.com/diverse-ensemble for the full appendix.

REFERENCES

[1] Gaon An, Seungyong Moon, Jang-Hyun Kim, and Hyun Oh Song. 2021. Uncertainty-based offline reinforcement learning with diversified Q-ensemble. In *Advances in neural information processing systems* (Virtual Event) *(NeurIPS 2021, Vol. 34)*. Curran Associates, Inc., Red Hook, NY, USA, 7436–7447.

[2] Kai Arulkumaran, Marc Peter Deisenroth, Miles Brundage, and Anil Anthony Bharath. 2017. Deep reinforcement learning: A brief survey. *IEEE Signal Processing Magazine* 34, 6 (2017), 26–38.

[3] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. 2016. OpenAI Gym. *arXiv preprint arXiv:1606.01540* preprint (2016), 4 pages.

[4] Tianqi Chen and Carlos Guestrin. 2016. XGBoost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (San Francisco, CA, USA) *(KDD '16)*. ACM, New York, NY, USA, 785–794.

[5] Karl Cobbe, Oleg Klimov, Chris Hesse, Taehoon Kim, and John Schulman. 2019. Quantifying generalization in reinforcement learning. In *Proceedings of the 36th International Conference on Machine Learning* (Long Beach, CA, USA) *(ICML 2019, Vol. 97)*. PMLR, Online, 1282–1289.

[6] Yan Duan, Xi Chen, Rein Houthooft, John Schulman, and Pieter Abbeel. 2016. Benchmarking deep reinforcement learning for continuous control. In *Proceedings of the 33rd International Conference on Machine Learning* (New York, NY, USA) *(ICML 2016, Vol. 48)*. PMLR, Online, 1329–1338.

[7] U.S. Department of Energy. 2006. *Benefits of demand response in electricity markets and recommendations for achieving them.* Technical Report. Lawrence Berkeley National Laboratory, Berkeley, CA, USA.

[8] Scott Fujimoto, Herke Hoof, and David Meger. 2018. Addressing function approximation error in actor-critic methods. In *Proceedings of the 35th International Conference on Machine Learning* (Stockholm, Sweden) *(ICML 2018, Vol. 80)*. PMLR, Online, 1582–1591.

[9] M.A. Ganaie, Minghui Hu, A.K. Malik, M. Tanveer, and P.N. Suganthan. 2022. Ensemble deep learning: A review. *Engineering Applications of Artificial Intelligence* 115 (2022), 105151.

[10] Dibya Ghosh, Jad Rahme, Aviral Kumar, Amy Zhang, Ryan P. Adams, and Sergey Levine. 2021. Why generalization in RL is difficult: Epistemic POMDPs and implicit partial observability. In *Advances in Neural Information Processing Systems* (Virtual Event) *(NeurIPS 2021, Vol. 34)*. Curran Associates, Inc., Red Hook, NY, USA, 25502–25515.

[11] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. 2018. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *Proceedings of the 35th International Conference on Machine Learning* (Stockholm, Sweden) *(ICML 2018, Vol. 80)*. PMLR, Online, 1329–1338.

[12] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. 2009. *The elements of statistical learning: Data mining, inference, and prediction* (2nd ed.). Springer Publishing, New York, NY, USA.

[13] A. Rezaee Jordehi. 2019. Optimisation of demand response in electric power systems, a review. *Renewable and Sustainable Energy Reviews* 103 (2019), 308–319.

[14] Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of the 3rd International Conference on Learning Representations* (San Diego, CA, USA) *(ICLR 2015)*. arXiv.org, Online, 11 pages.

[15] Robert Kirk, Amy Zhang, Edward Grefenstette, and Tim Rocktäschel. 2023. A survey of zero-shot generalisation in deep reinforcement learning. *Journal of Artificial Intelligence Research* 76 (2023), 201–264.

[16] Intelligent Environments Lab and AIcrowd. 2022. NeurIPS 2022 CityLearn challenge. https://www.aicrowd.com/challenges/neurips-2022-citylearn-challenge. Accessed: 2023-01-30.

[17] Kimin Lee, Michael Laskin, Aravind Srinivas, and Pieter Abbeel. 2021. SUNRISE: A simple unified framework for ensemble learning in deep reinforcement learning. In *Proceedings of the 38th International Conference on Machine Learning* (Virtual Event) *(ICML 2021, Vol. 139)*. PMLR, Online, 6131–6141.

[18] Chenghao Li, Tonghan Wang, Chengjie Wu, Qianchuan Zhao, Jun Yang, and Chongjie Zhang. 2021. Celebrating diversity in shared multi-agent reinforcement learning. In *Advances in Neural Information Processing Systems* (Virtual Event) *(NeurIPS 2021, Vol. 34)*. Curran Associates, Inc., Red Hook, NY, USA, 3991–4002.

[19] Timothy P. Lillicrap, Jonathan J. Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. 2016. Continuous control with deep reinforcement learning. In *Proceedings of the 4th International Conference on Learning Representations* (San Juan, Puerto Rico) *(ICLR 2016)*. OpenReview.net, Online, 10 pages.

[20] Zhuang Liu, Xuanlin Li, Bingyi Kang, and Trevor Darrell. 2021. Regularization matters in policy optimization–An empirical study on continuous control. In *Proceedings of the 9th International Conference on Learning Representations* (Virtual Event, Austria) *(ICLR 2021)*. OpenReview.net, Online, 12 pages.

[21] Muhammad A. Masood and Finale Doshi-Velez. 2019. Diversity-inducing policy gradient: Using maximum mean discrepancy to find a set of diverse policies. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence AI for Improving Human Well-being* (Macao, China) *(IJCAI 2019)*. International Joint Conferences on Artificial Intelligence Organization, Online, 5923–5929.

[22] Kevin R. McKee, Joel Z. Leibo, Charlie Beattie, and Richard Everett. 2022. Quantifying the effects of environment and population diversity in multi-agent reinforcement learning. *Autonomous Agents and Multi-Agent Systems* 36, 21 (2022), 16 pages.

[23] Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. 2016. Asynchronous methods for deep reinforcement learning. In *Proceedings of the 33rd International Conference on Machine Learning* (New York, NY, USA) *(ICML 2016, Vol. 48)*. PMLR, Online, 1928–1937.

[24] Nadim Nachar. 2008. The Mann-Whitney U: A test for assessing whether two independent samples come from the same distribution. *Tutorials in quantitative Methods for Psychology* 4, 1 (2008), 13–20.

[25] Kingsley Nweye, Siva Sankaranarayanan, and Zoltán Nagy. 2022. MERLIN: Multi-agent offline and transfer learning for occupant-centric energy flexible operation of grid-interactive communities using smart meter data and CityLearn. preprint (2022), 17 pages. arXiv:2301.01148

[26] Kingsley Nweye, Sankaranarayanan Siva, and Gyorgy Zoltan Nagy. 2023. The CityLearn Challenge 2022. https://doi.org/10.18738/T8/0YLJ6Q

[27] Jack Parker-Holder, Aldo Pacchiano, Krzysztof M. Choromanski, and Stephen J. Roberts. 2020. Effective diversity in population based reinforcement learning. In *Advances in Neural Information Processing Systems* (Virtual Event) *(NeurIPS 2020, Vol. 33)*. Curran Associates, Inc., Red Hook, NY, USA, 18050–18062.

[28] Lior Rokach. 2019. *Ensemble learning: Pattern classification using ensemble methods* (2nd ed.). Machine perception and artificial intelligence, Vol. 85. World Scientific Publishing Co. Pte. Ltd., Singapore.

[29] Omer Sagi and Lior Rokach. 2018. Ensemble learning: A survey. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* 8, 4 (2018), e1249.

[30] Masha Samsikova. 2021. Effective Diversity in Population Based Reinforcement Learning: DvD-TD3 Pytorch Implementation. https://github.com/holounic/DvD-TD3/. Accessed: 2023-01-30.

[31] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal Policy Optimization Algorithms. preprint (2017), 9 pages. arXiv:1707.06347

[32] Matthijs T.J. Spaan. 2012. *Partially observable Markov decision processes.* Springer Berlin Heidelberg, Berlin, Heidelberg, Chapter 12, 387–414.

[33] Richard S. Sutton and Andrew G. Barto. 2018. *Reinforcement learning: An introduction.* MIT press, Cambridge, MA, USA.

[34] José R. Vázquez-Canteli, Jérôme Kämpf, Gregor Henze, and Zoltán Nagy. 2019. CityLearn v1.0: An OpenAI Gym Environment for Demand Response with Deep Reinforcement Learning. In *Proceedings of the 6th ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation* (New York, NY, USA) *(BuildSys '19)*. ACM, New York, NY, USA, 356–357.

[35] José R. Vázquez-Canteli and Zoltán Nagy. 2019. Reinforcement learning for demand response: A review of algorithms and modeling techniques. *Applied Energy* 235 (2019), 1072–1089.

[36] Sam Witty, Jun K. Lee, Emma Tosch, Akanksha Atrey, Kaleigh Clary, Michael L. Littman, and David Jensen. 2021. Measuring and characterizing generalization in deep reinforcement learning. *Applied AI Letters* 2, 4 (2021), e45.

[37] Hongyang Yang, Xiao-Yang Liu, Shan Zhong, and Anwar Walid. 2020. Deep reinforcement learning for automated stock trading: An ensemble strategy. In *Proceedings of the First ACM International Conference on AI in Finance* (New York, NY, USA) *(ICAIF '20)*. ACM, New York, NY, USA, Article 31, 8 pages.

[38] Tianyu Zhang, Aakash Krishna G.S., Mohammad Afshari, Petr Musilek, Matthew E. Taylor, and Omid Ardakanian. 2022. Diversity for Transfer in Learning-Based Control of Buildings. In *Proceedings of the 13th ACM International Conference on Future Energy Systems* (Virtual Event) *(e-Energy '22)*. ACM, New York, NY, USA, 556–564.

[39] Zhi-Hua Zhou. 2012. *Ensemble methods: Foundations and algorithms* (1st ed.). CRC Press, Boca Raton, FL, USA.