Structure Detection for Contextual Reinforcement Learning

Tianyue Zhou*, Jung-Hoon Cho*, Cathy Wu

Massachusetts Institute of Technology
{tianyuez, jhooncho, cathywu}@mit.edu

Abstract

Contextual Reinforcement Learning (CRL) tackles the problem of solving a set of related Contextual Markov Decision Processes (CMDPs) that vary across different context variables. Traditional approaches—independent training and multi-task learning—struggle with either excessive computational costs or negative transfer. A recently proposed multi-policy approach, Model-Based Transfer Learning (MBTL), has demonstrated effectiveness by strategically selecting a few tasks to train and zero-shot transfer. However, CMDPs encompass a wide range of problems, exhibiting structural properties that vary from problem to problem. As such, different task selection strategies are suitable for different CMDPs. In this work, we introduce Structure Detection MBTL (SD-MBTL), a generic framework that dynamically identifies the underlying generalization structure of CMDP and selects an appropriate MBTL algorithm. For instance, we observe MOUNTAIN structure in which generalization performance degrades from the training performance of the target task as the context difference increases. We thus propose M/GP-MBTL, which detects the structure and adaptively switches between a Gaussian Process-based approach and a clustering-based approach. Extensive experiments on synthetic data and CRL benchmarks—covering continuous control, traffic control, and agricultural management—show that M/GP-MBTL surpasses the strongest prior method by 12.49% on the aggregated metric. These results highlight the promise of online structure detection for guiding source task selection in complex CRL environments.

Code — https://github.com/mit-wu-lab/SD-MBTL/
Webpage — https://mit-wu-lab.github.io/SD-MBTL/

1 Introduction

Despite the recent success of deep reinforcement learning (RL), deep RL often struggles to solve real-world applications that involve families of related tasks that differ in a few key parameters but mostly share underlying dynamics (Bellemare et al. 2020; Degrave et al. 2022; Benjamins et al. 2023). Such contextual variations naturally arise in robotics (e.g., different payload weights or terrain conditions) (Yu et al. 2020) and traffic control (e.g., varying traffic inflows or signal timing) (Jayawardana et al. 2025). Formally, these settings can be modeled as Contextual Markov Decision Processes (CMDPs) (Hallak, Di Castro, and Mannor 2015; Modi et al. 2018; Benjamins et al. 2023), where each task is specified by a context. Moreover, we extend prior studies to more challenging multi-dimensional CMDP, rather than a single-dimensional context.

Due to the curse of dimensionality in context variables, a central challenge in CMDPs is how to efficiently train policies that generalize to numerous related tasks without starting training from scratch. Existing paradigms for CMDPs include: (1) Independent training of a separate policy for each task, which is straightforward but computationally expensive for large task families; (2) Multi-task training of a single universal policy conditioned on the context, which can suffer from limited model capacity or negative transfer if tasks are too dissimilar (Kang, Grauman, and Sha 2011; Standley et al. 2020); and (3) Multi-policy training on a small subset of source tasks *with zero-shot transfer* to new tasks, raising the key question of how to choose that subset. Recent work has shown that carefully selecting which tasks to train on can lead to strong generalization and improved sample efficiency (Cho et al. 2023, 2024), since the cost of full training far outweighs the cost of policy evaluation and the task-selection.

Multi-policy training requires selecting a subset of tasks in CMDPs, formulated as a Greedy Tasks Selection Problem (GSTS). However, many CMDPs exhibit specific structural patterns that can be exploited. For example, (Cho et al. 2024) splits generalization performance into training performance and a generalization gap, modeling the gap as a linear function. Inspired by this, we define a task structure as a functional decomposition of generalization performance in which certain components satisfy designated properties. We thus propose a generic *Structure Detection Model-Based Transfer Learning* (SD-MBTL) framework, which detects the underlying structure and adapts its task selection strategy accordingly. In this paper, we instantiate the SD-MBTL framework with **M/GP-MBTL**, which uses two specific structures and dynamically switches between task selection algorithms–Gaussian Process-based and clustering-based MBTL (Figure 1).

For example, we observe experimentally that generalization in CMDPs often follow some structure, in which we find some task selection approach works better. Thus, detecting such structure enables a more targeted approach to source task selection than a one-size-fits-all approach. To capitalize

*These authors contributed equally.

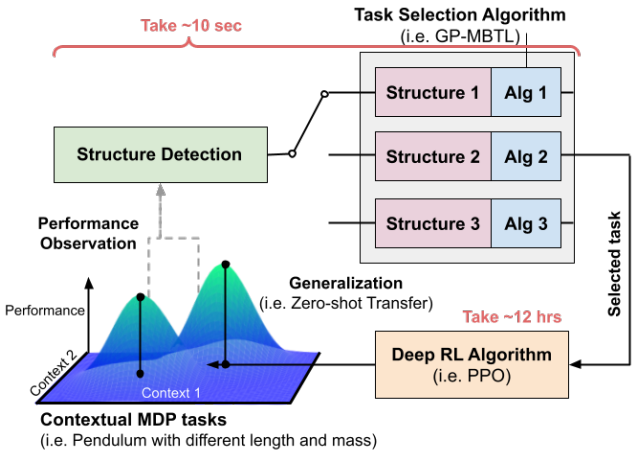


Figure 1: **Conceptual overview.** SD-MBTL detects an underlying structure from observed generalization performance and selects an appropriate algorithm.

on this insight, we also devise a fast and effective algorithm, MOUNTAIN Model-Based Transfer Learning (M-MBTL), by reducing GSTS problem to a sequential version of clustering problem upon a certain MOUNTAIN structure. It enables us to leverage clustering loss to efficiently guide the search of the training task in a continuous space.

Our main contributions are summarized as follows:

- We introduce SD-MBTL, a unified framework that detects the underlying generalization structure of CMDPs and adapts its source-task selection strategy, resulting in improved sample efficiency in diverse CRL settings.
- We propose M/GP-MBTL, a practical structure detection algorithm, using two specific structures in the generalization performance. We also develop a clustering-based MBTL algorithm for MOUNTAIN structure.
- We empirically validate our approach in a multi-dimensional synthesized dataset and real-world CMDP benchmarks, including continuous control (CartPole and BipedalWalker (Benjamins et al. 2023)), cooperative eco-driving (IntersectionZoo (Jayawardana et al. 2025)), and crop management (CyclesGym (Turchetta et al. 2022)). On the aggregated metric normalized between the random baseline and the oracle, M/GP-MBTL outperforms the previous best method by 12.49%.

The rest of the article is structured as follows: Section 2 reviews CMDP and GSTS problem. Section 3 introduces the SD-MBTL framework. Section 4 introduces M/GP-MBTL. Section 5 presents evaluations on benchmarks, Section 6 discusses related work, and Section 7 concludes.

2 Preliminaries

2.1 Contextual Markov Decision Process

We consider a *contextual* Markov decision process (CMDP) defined over a family of MDPs indexed by a *context* (or *task*) variable in a multi-dimensional space. Let the finite set of all MDPs (or tasks) be denoted by $Y = \{y_1, y_2, \ldots, y_N\}$, where each task $y_n \in \mathbb{R}^D$ for some dimension D, and $N = |Y|$ is the total number of tasks (or target tasks). Specifically, let each task $y \in Y \subset \mathbb{R}^D$ parameterize a distinct MDP $\mathcal{M}_y = (S, A, P_y, R_y, \rho_y)$, where S is the state space, A is the action space, P_y represents transition dynamics, R_y is the reward function, and ρ_y is the initial state distribution for the task y (Hallak, Di Castro, and Mannor 2015; Modi et al. 2018). Hence, the CMDP is the collection $\{\mathcal{M}_y\}_{y \in Y}$.

We define a continuous set of tasks $X \subset \mathbb{R}^D$ such that X has the same bound as Y. Intuitively, X captures an underlying continuous space of tasks from which Y can be viewed as a discrete subset ($Y \subset X$). Given a context $x \in X$, one can train a policy π_x (using, for instance, an off-the-shelf RL algorithm) and get the *(source) training performance* on task x by $J(\pi_x, x) = \mathbb{E}\big[\text{return of } \pi_x \text{ in } \mathcal{M}_x\big]$. To zero-shot transfer from a source task x to a target task y, we apply the policy π_x (trained on $\mathcal{M}_x$) within $\mathcal{M}_y$, potentially at reduced performance compared to training directly on y. The resulting *generalization performance* is $J(\pi_x, y)$. We quantify the zero-shot generalization gap when transferring from x to y as $\Delta J(\pi_x, y) = J(\pi_x, x) - J(\pi_x, y)$.

2.2 Problem Formulation

Greedy source task selection (GSTS) problem[1] Given a CMDP with a finite set of *target tasks* Y, our goal is to identify a sequence of tasks on which to perform full RL training, so as to achieve high generalization performance on all tasks in Y. Specifically, at each step k, we pick a new source task $x_k \in X$ to train a corresponding policy π_{x_k}. For each task in Y, we choose the best performing policy from a set of trained policies, which maximizes the performance. The objective at each round k is to *greedily* select a new source task x_k that maximizes the expected performance across the entire set Y. One can write this as:

$$x_k = \underset{x \in X \setminus \{x_{1:k-1}\}}{\arg\max} \; \mathbb{E}_{y \sim \mathcal{U}(Y)}\Big[\max_{x' \in x_{1:k-1} \cup \{x\}} J(\pi_{x'}, y)\Big] \quad (1)$$

where $x_{1:K} = x_1, \ldots, x_K$, and $\mathcal{U}(Y)$ is a uniform distribution over the finite task set Y. In general, an exact solution to GSTS problems can be intractable for large-scale or continuous task spaces. In practice, many applications have fairly small task spaces (e.g., 3-10 context dimensions, each discretized to 100 values). The core challenge is that solving *any* task MDP is fairly expensive and thus the number of task MDPs to solve should be minimized. MBTL aimed to circumvent naive exploration by modeling the training performance and generalization gap and optimizing the selection via Bayesian optimization (Cho et al. 2024).

2.3 Model-Based Transfer Learning

A practical solution to solve GSTS problem is provided by MBTL (Cho et al. 2024), which we refer to as **GP-MBTL** in this work. GP-MBTL discretizes the continuous source-task

[1]This was defined as the Sequential Source Task Selection problem in (Cho et al. 2024). We renamed the problem because its goal is to greedily select a training task at each step, and the term "sequential" fails to convey this aspect.

space, models source-task returns with Gaussian-Process regression, and approximates the generalization gap as a linear function of context similarity. At each iteration k, it selects the source task x_k that maximizes an acquisition function combining predicted source task performance $\mu_{k-1}(x)$, uncertainty $\sigma_{k-1}(x)$, and the estimated generalization gap to target tasks. The policy trained on x_k is then evaluated, and the GP posterior updated, progressively improving coverage of the target set Y. We additionally use observed transfer performance to refine the acquisition rule and estimate the gap's slope online. Full derivations, algorithmic details, and hyperparameters are provided in Appendix I.

3 Structure Detection Model-Based Transfer Learning

CMDPs with multi-dimensional context spaces typically demand significantly more data to keep model predictions accurate, which challenges the Gaussian-process module in GP-MBTL and complicates the exploration–exploitation tradeoff. When a CMDP obeys a recognisable structure, however, that structure can be detected and then exploited to curb unnecessary exploration. We therefore introduce a generic **Structure Detection Model-Based Transfer Learning** (SD-MBTL) framework for task selection.

3.1 Generalization Performance Structure Decomposition

To support the identification and analysis of CMDP structures, we decompose the generalization-performance structure inspired by the Sobol–Hoeffding (functional-ANOVA) decomposition, which uniquely and orthogonally splits any square-integrable multivariate function into additive main-effect terms and interaction terms (Hoffding 1948; Sobol' 1990). More details on how we derive this decomposition are provided in Appendix C.

Definition 3.1 (Generalization Performance Structure Decomposition)**.** For any task pair (x, y), define $C := \mathbb{E}_{x\in X, y\in Y}[J(\pi_x, y)]$, $g(y) := \mathbb{E}_{x\in X}[J(\pi_x, y)] - C$, $f(x) := J(\pi_x, x) - g(x) - C$, and $h(x, y) := J(\pi_x, y) - f(x) - g(y) - C$, thus we can get:

$$J(\pi_x, y) = f(x) + g(y) + h(x, y) + C, \tag{2}$$

and $h(x, y) = 0$ if $x = y$.

Based on empirical observations, we found three components that affect generalization performance: 1) The intrinsic quality of source policy influences whether it has good generalization across many tasks. 2) The difficulty of the target task affects whether transferred policies can achieve high performance on it. 3) The dissimilarity between source and target tasks degrades generalization performance, as large differences in context usually yield poor generalization. Thus, we name $f(x)$ as the **policy quality**, $g(y)$ as the **task difficulty**, and $h(x, y)$ as the **task dissimilarity** between source and target tasks.

Based on the decomposition, we outline the procedure for addressing a CMDP by exploiting a specific structure. Within a CMDP structure, one or more of $f(x)$, $g(y)$, and $h(x, y)$ may satisfy particular assumptions or properties, allowing us to treat them as known components of $J(\pi_x, y)$. Thus, a corresponding algorithm can be used to learn the remaining unknown components and select training tasks based on the model so as to maximize the objective.

3.2 SD-MBTL

SD-MBTL embeds a dynamic structure detection mechanism inside the MBTL loop and chooses the corresponding MBTL algorithm based on the detected structure. Its inputs are (i) a set of candidate CMDP structures $\mathcal{S}$, (ii) a detection routine $\text{Detect} : \mathbb{R}^{k\times N} \to \mathcal{S}$, and (iii) a library of MBTL algorithms $\{\text{Alg}_1, \ldots, \text{Alg}_{|\mathcal{S}|}\}$ matched one-to-one with the structures in $\mathcal{S}$. At each decision round, SD-MBTL uses the observed generalization performances $\{J(\pi_{x_\kappa}, y_n)\}_{n\in[N],\, \kappa\in[k]}$ to infer the current structure $s_i = \text{Detect}(\cdot)$, then invokes the corresponding algorithm Alg_i to choose the next training task x. After training, the new policy π_x is transferred to all target tasks, yielding zero-shot generalization performances $\{J(\pi_x, y_n)\}_{n\in[N]}$. More details in SD-MBTL are provided in Appendix K.

4 M/GP Model-Based Transfer Learning

As a concrete instantiation of SD-MBTL, we propose **M/GP-MBTL**, which targets a special CMDP structure—MOUNTAIN. When the CMDP satisfies MOUNTAIN, the GSTS problem reduces to clustering, and we employ the clustering-based M-MBTL. Otherwise, we revert to the more general GP-MBTL. This two-way strategy combines the sample efficiency of M-MBTL with the robustness of GP-MBTL.

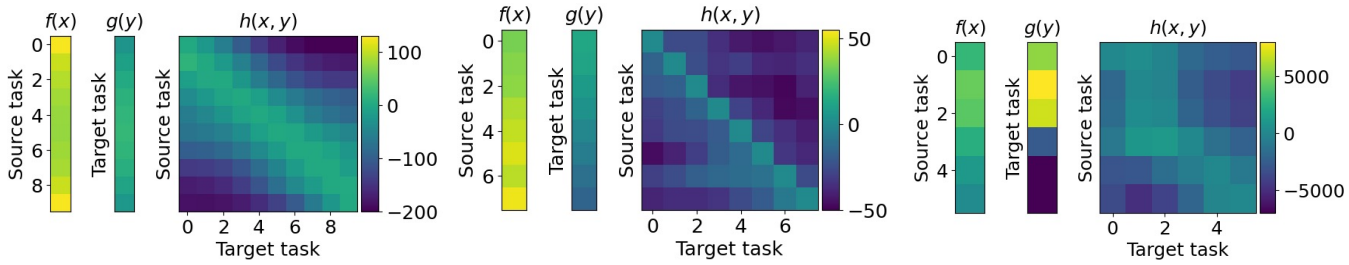
4.1 MOUNTAIN Structure

In our experiments, we observe that some CMDP problems exhibit similar model structures. Figure 2 shows the policy quality $f(x)$, task difficulty $g(y)$, and task dissimilarity $h(x, y)$ in CartPole, BipedalWalker, and CyclesGym benchmarks. In these tasks, $f(x)$ is nearly constant. $h(x, y)$ decreases approximately linearly as the context difference increases, resembling a distance metric. These similar characteristics suggest that they may originate from the same underlying structure. Thus, we identify a specific generalization structure—termed MOUNTAIN structure—that relies on the following key assumptions.

Assumption 4.1 (Constant Source Task Influence)**.** Policy quality is constant: $f(x) = C_1$.

Assumption 4.2 (Distance Metric for Task Dissimilarity)**.** The task dissimilarity $h(x, y)$ is represented as a distance metric: $h(x, y) = -\text{dist}(x, y)$. The distance metric (Čech and Katětov 1969) is a function $\text{dist} : \mathbb{R}^D \times \mathbb{R}^D \to \mathbb{R}$ which satisfies $\text{dist}(x, x) = 0\ \forall x$, positivity, symmetry, and the triangle inequality. In this paper, we use the L_1 norm as the distance metric for our algorithms.

The linear generalization gap assumption in (Cho et al. 2023) and (Cho et al. 2024) assumes $J(\pi_x, y) - J(\pi_x, x) = g(x) - g(y) + h(x, y) = k|x - y|$. It can be considered a specific case of Assumption 4.2, where the CMDP is one-dimensional and target task difficulties $g(y)$ are constant.



(a) CartPole (b) BipedalWalker (c) CyclesGym

Figure 2: Heatmap of policy quality $f(x)$, task difficulty $g(y)$, and task dissimilarity $h(x, y)$ for CartPole (mass of cart), BipedalWalker (scale), and CyclesGym (precipitation) averaged over three different random seeds. In these tasks, $f(x)$ is nearly constant. $h(x, y)$ decreases approximately linearly as the context difference increases, resembling a distance metric.

Definition 4.3 (MOUNTAIN Structure)**.** MOUNTAIN is a CMDP structure where Assumption 4.1 and 4.2 holds. Or equivalently, $J(\pi_x, y) = J(\pi_y, y) - \text{dist}(x, y)$.

The proof of equivalence is provided in Appendix D. We refer to this as MOUNTAIN because the training performance of the target task resembles mountain peaks, and the generalization performance acts like a slope that decreases as the distance between source and target tasks increases. This perspective effectively reduces GSTS to a clustering problem, which will be discussed in Section 4.3.

4.2 Structure Detection

For an unknown CMDP, we aim to determine whether it satisfies MOUNTAIN structure. At each step k we have observed the generalization performances $J(\pi_x, y)$ for all $x \in \{x_1, \ldots, x_k\}$. Our goal is to test whether $f(x)$ and $\text{dist}(x, y)$ individually obey Assumptions 4.1 and 4.2. Because MOUNTAIN structure does not include an explicit task difficulty term $g(y)$, we first remove the influence of task difficulty on $J(\pi_x, y)$ when performing structure detection. Then we apply two criteria: (i) a Small Variance Criterion to check whether policy quality is almost constant (Assumption 4.1), and (ii) a Slope Criterion to check that the majority of slopes are decreasing like a distance metric (Assumption 4.2). If both criteria are satisfied, we tag the CMDP as MOUNTAIN.

Removing the influence of task difficulty. For each step k, we estimate the difficulty of target task by the mean generalization performance of trained policies.

$$g(y) \approx \mathbb{E}_{x \in x_{1:k}}[J(\pi_x, y)] - C. \tag{3}$$

This estimate replaces $x \in X$ in the Definition 3.1 with $x \in x_{1:k}$, because at step k we only have access to the information in $x_{1:k}$. For any training task set $x_{1:k}$ at step k, we define the relative generalization performance as the difference between real generalization performance and its expectation over training tasks: $\overline{J}(\pi_x, y) := J(\pi_x, y) - \mathbb{E}_{x' \in x_{1:k}}[J(\pi_{x'}, y)]$. By removing the influence of task difficulty $g(y)$ by Equation 3, we can have $\overline{J}(\pi_x, y) \approx f(x) + h(x, y)$.

Small Variance Criterion. Although Assumption 4.1 treats policy quality $f(x)$ as a constant, a small variance in $f(x)$ does not materially distort the structure. We therefore compare the standard deviations of $f(x)$ with task dissimilarity $h(x, y)$ to verify that $f(x)$ indeed exhibits small deviation.

Lemma 4.4 (Relative Influence of Policy Quality and Task Dissimilarity)**.** *If* $std_{x \in x_{1:k}}(\overline{J}(\pi_x, x)) < \mathbb{E}_{x \in x_{1:k}}[std_{y \in Y}(\overline{J}(\pi_x, y))]$*, then we have* $std_{x \in x_{1:k}}(f(x)) < \mathbb{E}_{x \in x_{1:k}}[std_{y \in Y}(h(x, y))]$.

When Lemma 4.4 holds, policy quality $f(x)$ has a relatively small variance. The proof is provided in Appendix E. Define $\mathbb{I}(\cdot)$ as the indicator function. Thus, for any training task set $x_{1:k}$ at step k, we have the following criterion:

$$\begin{aligned}&\text{Small Variance Criterion}\\ :=&\mathbb{I}\left(\text{std}_{x \in x_{1:k}}(\overline{J}(\pi_x, x)) < \mathbb{E}_{x \in x_{1:k}}[\text{std}_{y \in Y}(\overline{J}(\pi_x, y))]\right).\end{aligned} \tag{4}$$

Slope Criterion. A distance metric $\text{dist}(x, y)$ shows both downward slopes on both sides of the source task, while Assumption 4.2 can be violated by the case where $h(x, y)$ have different slope signs. To capture this, we regress the observed generalization performance on signed context differences and extract the left- and right-hand slope vectors $\theta_{\text{L}}, \theta_{\text{R}} \in \mathbb{R}^D$. We verify this structure by checking whether the majority of slopes are indeed decreasing, which is equivalent to confirming that the slope signs on both sides are the same, excluding cases where both sides slope upward. Thus, we have:

$$\text{Slope criterion} := \mathbb{I}\left(\frac{1}{D}\sum_{d=1}^{D}\mathbb{I}[\text{sgn}(\theta_L^d) = \text{sgn}(\theta_R^d)] > \frac{1}{2}\right), \tag{5}$$

where θ_{L}^d and θ_{R}^d represents the slope in the d-th dimension. More details are provided in Appendix G. We define $\text{Detect}(\{J(\pi_\kappa, y_n)\}_{n \in [N], \kappa \in [k-1]})$ = MOUNTAIN if both criteria are satisfied; otherwise, the result is NONE.

4.3 MOUNTAIN Model-Based Transfer Learning

Under MOUNTAIN, GSTS problem is reduced to a sequential clustering problem (Lemma 4.5). Proof is provided in Appendix F. Specifically, the reduced problem is a sequential minimization of the total distance between target tasks and their corresponding source tasks. Thus, a clustering loss function can be used to search for training tasks fast and accurately in a continuous set X.

Algorithm 1: M/GP-MBTL

Input: Task set $Y = \{y_n\}_{n\in[N]}$
for $k \in [K]$ **do**
  $s = \text{Detect}(\{J(\pi_\kappa, y_n)\}_{n\in[N],\kappa\in[k-1]})$
  if $s =$MOUNTAIN **then**
    $x_k \leftarrow$ Run M-MBTL
  else
    $x_k \leftarrow$ Run GP-MBTL
  end if
  Train on x_k, receive $\{J(\pi_k, y_n)\}_{n\in[N]}$
end for

Lemma 4.5 (Reduction of GSTS to Clustering)**.** *With the* MOUNTAIN *structure, GSTS problem reduces to the sequential version of the clustering problem. Thus, a clustering loss function can be used to search for training tasks fast and accurately in a continuous set X.*

$$\textit{for } k \in [K]: \; x_k = \arg\min_x \mathbb{E}_{y_n}[\min_{x'\in x_{1:k-1}\cup\{x\}} \textit{dist}(x', y_n)] \tag{6}$$

Our MOUNTAIN Model-Based Transfer Learning (M-MBTL) extends K-Means to this sequential setting. Because fixing earlier centroids can trap the search in local optima, we adopt a random restart technique (Yagiura and Ibaraki 2001): sample M target tasks as initial centroids, locally refine each by the clustering loss above, and choose the one with the lowest loss as the training task for that round. We repeat this procedure for K rounds, training a policy on each selected task and evaluating it on all targets. Full algorithms, including our techniques to reduce the time complexity, are provided in Appendix H.1. We also provide a comparison between M-MBTL and GP-MBTL in Appendix J.

4.4 M/GP-MBTL

Based on the two algorithms and the detection approach above, we introduce a concrete instantiation of SD-MBTL, named M/GP-MBTL (Algorithm 1). This algorithm uses the function Detect$(\cdot)$ mentioned in Section 4.2 to detect the problem structure. Then it dynamically chooses between M-MBTL for MOUNTAIN structures and GP-MBTL for more general settings, enabling the algorithm to tailor its task selection strategy to the underlying CMDP structure.

5 Experiments

In this section, we present an extensive evaluation of our approaches, organized around two guiding questions: **Q1**: Under what conditions does M-MBTL perform well, and under what conditions does GP-MBTL perform well? **Q2**: Can M/GP-MBTL always achieve the best of both worlds, or approach optimal performance? We investigate these questions on CMDP experiments, including synthetic data, continuous control, eco-friendly traffic control, and crop management. Appendix O.3 reports ablation studies to assess how different combinations of algorithms in SD-MBTL affect performance, and Appendix H.3 compares the runtime. Appendix O.2 shows the algorithm selection at each round and the number of training policies required to achieve ϵ-suboptimal.

5.1 Setup

Although our aim is to tackle high-dimensional CMDPs, the number of tasks—and hence the training time—grows exponentially with dimensionality. Therefore, we confine our experiments to three-dimensional settings.

Baselines. We consider two types of baselines: canonical and multi-policy. The canonical baselines include: (1) **Independent training**, which trains separate policies on each MDP task; and (2) **Multi-task training**, where a universal context-conditioned policy is trained for all tasks. The multi-policy baselines include: (3) **Random selection**, which selects training tasks uniformly at random; (4) **GP-MBTL** (Cho et al. 2024); (5) **M-MBTL**; and (6) **Myopic Oracle**, an optimal solution for GSTS problem which has access to the generalization performance of each source-target pair (x, y) in all experiment trials and selects the training task myopically in each step. More details, including the formal definition and relationship to prior work, are provided in Appendix L.

Train and zero-shot transfer. We discretize the continuous task set X into N tasks. For the ease of our problem, we set these discretized tasks the same as Y. We use Proximal Policy Optimization (PPO) (Schulman et al. 2017) for training. We zero-shot transfer the policies obtained after training each task to all other tasks. The generalization performance constructs a transfer matrix whose rows represent source tasks and columns represent target tasks. We set the number of decision rounds $K = N$, and train three times with different random seeds. More details on deep RL implementation and hyperparameters are provided in Appendix N.

Bootstrapping. However, three matrices are not enough for evaluating our different source task selections. To address this, we employ *bootstrapping* (Tibshirani and Efron 1993) to synthesize an expanded dataset. Since each task is independently trained three times, the rows of the transfer matrix are independent. Therefore, we generate 100 bootstrapped transfer matrices with different random seeds by resampling rows of the transfer matrices with replacement from the original set of trained policies. Through bootstrapping, each row of the transfer matrix is still sampled from its true distribution, but the number of transfer matrices is significantly increased. It allows us to more reliably approximate the real distribution of the transfer matrix and thereby mitigate the effects of limited training runs.

Performance measure. We apply min-max normalization to the performance across all tasks after the algorithm is run. We use the mean as the performance metric, and use bootstrap resampling to compute the 95% confidence intervals (CI) of the performance. Because the upper and lower bounds of the CI are roughly symmetric, we only display $\pm\frac{(\text{upper}-\text{lower})}{2}$ as the CI half-width. We also evaluated the IQM and the median of the performance, which are presented in Appendix O.2. Both of them are close to the mean. Additionally, to compare methods across different benchmarks and cases, we use an **aggregated performance** metric adapted from the human-normalized score (Badia et al. 2020; Mnih et al. 2015): each algorithm's score on a benchmark is linearly rescaled so that the Random baseline maps to 0 and the Myopic Oracle

to 1, and we then average these normalized scores over all benchmarks. This single 0–1 value shows both how much an algorithm surpasses Random and how close it comes to the oracle; full definitions and formulas appear in Appendix M.

5.2 Synthetic Data

Data description. We first evaluate the performance of M/GP-MBTL under synthesized dataset, including the cases where Assumptions 4.1 and 4.2 are satisfied or not, as well as cases with varying noise levels. We discretize the three dimensional continuous task set X into $N = 8 \times 8 \times 8$ tasks, each characterized by three context dimensions, where each dimension is an integer ranging from 1 to 8. We generate $J(\pi_x, y) = f(x)+g(y)+h(x, y)+C+\epsilon_{x,y}$, where $C = 500$ is a constant that does not affect any algorithmic decisions, $\epsilon_{x,y} \sim \mathcal{N}(0, \sigma^2)$ is a Gaussian noise sampled independently for all x and y, and $\sigma = 5$. We include eight conditions, which is the combination of three variations: 1) constant and non-constant $f(x)$; 2) constant and non-constant $g(y)$; 3) distance metric holds or does not hold. We use a linear function $f(x) = [4, 4, 4]\cdot x$ and $g(y) = [3, 3, 3]\cdot y$ for the non-constant case, and $f(x) = 0$, $g(y) = 0$ for constant case. When the distance metric holds, $h(x, y) = -[3, 3, 3]\cdot|x-y|$, where $|\cdot|$ means the absolute value for each element in a vector. Otherwise, $h(x, y) = -([3, 3, 3]\cdot[x-y]_+ + [1, 1, -3]\cdot[x-y]_-)$. We also tested our approaches on the synthetic data with different noise levels in Appendix O.1.

Results. Table 1 presents the performance comparison on synthetic data. Rows shaded in light gray indicate the synthetic settings that satisfy our proposed MOUNTAIN structure assumption. **Answer to Q1**: When MOUNTAIN structure holds, M-MBTL achieves the best performance among multi-policy baselines and closely approaches the performance of the myopic oracle. This indicates that M-MBTL effectively exploits this structure to get near-optimal performance with reduced training cost. Conversely, when MOUNTAIN structure is violated, GP-MBTL achieves the highest performance, demonstrating a general ability to handle CMDP problems. **Answer to Q2**: Overall, M/GP-MBTL attains the best aggregated performance, demonstrating that it can successfully detect whether a CMDP satisfies MOUNTAIN structure and choose the appropriate algorithm accordingly.

5.3 CMDP Benchmarks

Benchmarks. We consider four CMDP benchmarks: CartPole, BipedalWalker, IntersectionZoo, and CyclesGym. In CartPole, the objective is to keep a pole balanced upright on a moving cart. We vary three key context variables—length of the pole, mass of the pole, and mass of the cart—in ranges spanning from 0.2 to 2 times their standard values ($N = 9 \times 10 \times 10$). BipedalWalker is a 4-joint walker robot environment. Context variables are friction, gravity, and scale, ranging from 0.2 to 1.6 times their default values ($N = 8 \times 8 \times 8$). We use CARL environments (Benjamins et al. 2023) for these CartPole and BipedalWalker. IntersectionZoo (Jayawardana et al. 2025) is a multi-agent CRL benchmark for eco-driving in urban road networks. We run experiments on a synthetic intersection network configuration with three context variables—traffic inflow, autonomous vehicle (AV) penetration rate, and traffic signal green-phase duration—from 0.2 to 1.2 times their default values ($N = 6 \times 6 \times 6$), providing a broad range of complex traffic scenarios for evaluation. CyclesGym (Turchetta et al. 2022) is an agricultural management simulation environment in which an RL agent is tasked with managing various crop-related parameters. Context variables are temperature, sunlight, and precipitation. These variables are discretized into a $6 \times 6 \times 6$ grid, yielding different agricultural scenarios. More details on CMDP benchmarks are provided in Appendix N.2.

Results. Table 2 summarizes the results across four benchmarks. **Answer to Q1**: On the whole, the gap between the Myopic Oracle and canonical baselines (independent and multi-task training) indicate the potential of MBTL-based strategic task selection. Multi-task training performs the best on CartPole, which may be because the problem is relatively easy, such that a single straightforward context-conditioned policy can achieve strong performance across the CMDP. However, it performs worse in other benchmarks, potentially due to increased task complexity, where it may suffer from model capacity limitations. By contrast, M-MBTL and GP-MBTL show better aggregated metric performance than independent training and multi-task training. In tasks that satisfy MOUNTAIN structure (BipedalWalker, CyclesGym), M-MBTL achieves the highest performance. When the structure is violated (IntersectionZoo), GP-MBTL becomes superior, showing the value of a GP model for source-task performance. In addition, CartPole satisfies the assumptions of both M-MBTL and GP-MBTL, so both algorithms perform well. **Answer to Q2**: Across all four benchmarks, M/GP-MBTL consistently matches the stronger of M-MBTL and GP-MBTL. It achieves the best aggregated performance and shows an improvement of 0.1249 over GP-MBTL, indicating that it moves 12.49% closer to the Myopic Oracle within the MBTL framework. This highlights the method's effectiveness in addressing various CMDP problems.

6 Related Works

Contextual Reinforcement Learning and Multi-Policy Approaches. CRL provides a framework for managing variations of MDP in environment dynamics, rewards, and initial states through a context variable, effectively yielding a family of related tasks parameterized by these contexts (Hallak, Di Castro, and Mannor 2015; Modi et al. 2018; Benjamins et al. 2023). When the context is observed, a popular approach is to train a single policy that explicitly conditions on this contextual information (Teh et al. 2017; Sodhani, Zhang, and Pineau 2021) or by encoding it with a latent representation (Yu et al. 2020; Sun et al. 2022). Although performance can be strong when the tasks are homogeneous, negative transfer and capacity limits arise when context variation is large. An alternative is to train a small set of expert policies. This idea has been explored through policy committees (Ge et al. 2025) and represented-MDPs (Ivanov and Ben-Porat 2024), where the primary goal is to cluster the task space upfront and train one expert policy per cluster. Recent work shows that training only a few carefully chosen source tasks can outperform both independent training and multi-task training (Cho et al. 2023, 2024). We generalize these

$f(x)$	$g(y)$	$h(x,y)$	Random	GP-MBTL	M-MBTL (Ours)	M/GP-MBTL (Ours)	Myopic Oracle
Constant	Linear	Non-distance	0.6956 ± 0.0009	**0.7212** ± 0.0006	0.6976 ± 0.0002	0.7168 ± 0.0010	0.7260 ± 0.0001
Constant	Linear	L_1 norm	0.7011 ± 0.0002	0.6961 ± 0.0011	**0.7038** ± 0.0002	**0.7038** ± 0.0002	0.7048 ± 0.0002
Constant	None	Non-distance	0.7906 ± 0.0014	**0.8369** ± 0.0003	0.7936 ± 0.0003	0.8334 ± 0.0004	0.8375 ± 0.0002
Constant	None	L_1 norm	0.7722 ± 0.0004	0.7747 ± 0.0005	**0.7762** ± 0.0003	**0.7762** ± 0.0003	0.7778 ± 0.0003
Linear	Linear	Non-distance	0.5773 ± 0.0015	**0.6088** ± 0.0001	0.5782 ± 0.0001	**0.6086** ± 0.0001	0.6090 ± 0.0001
Linear	Linear	L_1 norm	0.5880 ± 0.0020	**0.6218** ± 0.0001	0.5900 ± 0.0002	**0.6216** ± 0.0002	0.6221 ± 0.0001
Linear	None	Non-distance	0.6672 ± 0.0020	**0.7083** ± 0.0002	0.6683 ± 0.0002	0.7073 ± 0.0006	0.7088 ± 0.0002
Linear	None	L_1 norm	0.7401 ± 0.0021	**0.7744** ± 0.0012	0.7422 ± 0.0002	**0.7751** ± 0.0002	0.7756 ± 0.0001
Aggregated Performance			-0.0000 ± 0.0107	0.2127 ± 0.0511	0.0147 ± 0.0089	**0.3099** ± 0.0290	1.0000 ± 0.0088

Table 1: Performance comparison on Synthetic Data ($\epsilon = \mathcal{N}(0, 5^2)$) with $K = 50$. Values are reported as the mean performance ± half the width of the 95% confidence interval. Rows shaded in gray indicate the settings that satisfy our proposed MOUNTAIN structure. Bold values represent the highest value(s) within the statistically significant range for each task, excluding the oracle.

Benchmark (CMDP)	Independent	Multi-task	Random	GP-MBTL	M-MBTL (Ours)	M/GP-MBTL (Ours)	Myopic Oracle
CartPole	0.9346	**0.9967**	0.9861	0.9919	0.9896	0.9898	0.9998
($K = 12$)	± 0.0003	± 0.0024	± 0.0017	± 0.0013	± 0.0016	± 0.0016	± 0.0000
BipedalWalker	0.7794	0.5680	0.8051	0.8073	**0.8315**	**0.8261**	0.8629
($K = 12$)	± 0.0011	± 0.0919	± 0.0045	± 0.0044	± 0.0029	± 0.0030	± 0.0011
IntersectionZoo	0.2045	0.3788	0.5288	**0.5840**	0.4878	**0.5682**	0.6305
($K = 50$)	± 0.0008	± 0.1059	± 0.0108	± 0.0092	± 0.0064	± 0.0069	± 0.0082
CyclesGym	0.2133	0.2081	0.2198	0.2193	**0.2205**	0.2201	0.2214
($K = 50$)	± 0.0002	± 0.0002	± 0.0001	± 0.0001	± 0.0001	± 0.0001	± 0.0001
Aggregated Performance	-2.9063 ± 0.1470	-3.1120 ± 1.8408	0.0000 ± 0.0467	0.1681 ± 0.0539	0.1822 ± 0.0503	**0.2930** ± 0.0403	1.0000 ± 0.0236

Table 2: Performance comparison of different methods on CMDP benchmarks. Values are reported as the mean performance ± half the width of the 95% confidence interval. Bold values represent the highest value(s) within the statistically significant range for each task, excluding the oracle.

ideas in multi-dimensional contexts and a *meta* layer that first detects what generalization structure is present before choosing which task-selection to deploy.

Structure Detection. Structure detection appears widely in machine learning, for example in detecting convexity for optimization problems (Ahmadi et al. 2013) and discovering GP kernels for nonparametric regression (Duvenaud et al. 2013). Typical frameworks include finding good approximations or decomposing the unknown functions. However, structure detection is fairly nascent in the context of CRL. In CRL, structure detection appears appears as (i) *context change detection*, where hierarchical RL agents infer when environment dynamics have shifted and switch options accordingly (Yücesoy and Tümer 2015); (ii) *model selection*, where online statistical tests reject over-complex dynamics models to curb over-fitting (Lee et al. 2021); and (iii) *policy-reuse selection*, where an agent leverages a pre-existing library of policies. These methods decide whether a source policy can be safely transferred to a new target task (Fernández and Veloso 2006) or combine multiple source policies to solve a target task, for example, through a mixture-of-experts model (Gimelfarb, Sanner, and Lee 2021). Our work also detects structure online, but we leverage it to choose a source-task, rather than to re-weight or gate a fixed library of policies; in principle, both ideas can be combined.

Appendix A surveys additional literature on clustering methods and solutions for CRL.

7 Conclusion

We propose a structure detection framework for CRL that infers the underlying structure of CMDPs and adaptively selects the task selection strategy. Our M/GP-MBTL algorithm switches between a clustering strategy for MOUNTAIN cases and a GP-based strategy otherwise. Experiments on synthetic and real benchmarks show consistent improvements over prior methods, highlighting the promise of structure detection for scalable and robust transfer learning in complex CRL environments. Nevertheless, our study has several **limitations**. First, the current detector focuses on a single structure; Second, structure inference relies on transfer evaluations of training tasks, which can be costly when the context space is very large or when policies are expensive to evaluate; Third, all experiments were conducted in three-dimensional context spaces; additional work is needed to confirm scalability to higher-dimensional settings. Future work will explore richer CMDP structures and develop specialized algorithms to reduce the number of required source-task trainings while maintaining (or improving) generalization performance.

Acknowledgements

This work was supported by the National Science Foundation (NSF) CAREER award (#2239566) and the Kwanjeong Educational Foundation Ph.D. scholarship program. The authors would like to thank the anonymous reviewers for their valuable feedback.

References

Ahmadi, A. A.; Olshevsky, A.; Parrilo, P. A.; and Tsitsiklis, J. N. 2013. NP-hardness of deciding convexity of quartic polynomials and related problems. *Mathematical programming*, 137: 453–476.

Badia, A. P.; Piot, B.; Kapturowski, S.; Sprechmann, P.; Vitvitskyi, A.; Guo, D.; and Blundell, C. 2020. Agent57: outperforming the Atari human benchmark. In *Proceedings of the 37th International Conference on Machine Learning*, ICML'20. JMLR.org.

Bellemare, M. G.; Candido, S.; Castro, P. S.; Gong, J.; Machado, M. C.; Moitra, S.; Ponda, S. S.; and Wang, Z. 2020. Autonomous navigation of stratospheric balloons using reinforcement learning. *Nature*, 588(7836): 77–82.

Benjamins, C.; Eimer, T.; Schubert, F.; Mohan, A.; Döhler, S.; Biedenkapp, A.; Rosenhahn, B.; Hutter, F.; and Lindauer, M. 2023. Contextualize Me – The Case for Context in Reinforcement Learning. *Transactions on Machine Learning Research*.

Cho, J.-H.; Jayawardana, V.; Li, S.; and Wu, C. 2024. Model-Based Transfer Learning for Contextual Reinforcement Learning. In *Advances in Neural Information Processing Systems*, volume 37, 88279–88319. Curran Associates, Inc.

Cho, J.-H.; Li, S.; Kim, J.; and Wu, C. 2023. Temporal Transfer Learning for Traffic Optimization with Coarse-grained Advisory Autonomy. arXiv:2312.09436.

Degrave, J.; Felici, F.; Buchli, J.; Neunert, M.; Tracey, B.; Carpanese, F.; Ewalds, T.; Hafner, R.; Abdolmaleki, A.; de Las Casas, D.; et al. 2022. Magnetic control of tokamak plasmas through deep reinforcement learning. *Nature*, 602(7897): 414–419.

Duvenaud, D.; Lloyd, J.; Grosse, R.; Tenenbaum, J.; and Zoubin, G. 2013. Structure discovery in nonparametric regression through compositional kernel search. In *International Conference on Machine Learning*, 1166–1174. PMLR.

Fernández, F.; and Veloso, M. 2006. Probabilistic policy reuse in a reinforcement learning agent. In *Proceedings of the fifth international joint conference on Autonomous agents and multiagent systems*, 720–727.

Ge, L.; Lanier, M.; Sarkar, A.; Guresti, B.; Zhang, C.; and Vorobeychik, Y. 2025. Learning Policy Committees for Effective Personalization in MDPs with Diverse Tasks. In *Forty-second International Conference on Machine Learning*.

Gimelfarb, M.; Sanner, S.; and Lee, C.-G. 2021. Contextual policy transfer in reinforcement learning domains via deep mixtures-of-experts. In de Campos, C.; and Maathuis, M. H., eds., *Proceedings of the Thirty-Seventh Conference on Uncertainty in Artificial Intelligence*, volume 161 of *Proceedings of Machine Learning Research*, 1787–1797. PMLR.

Hallak, A.; Di Castro, D.; and Mannor, S. 2015. Contextual markov decision processes. *arXiv preprint arXiv:1502.02259*.

Hoffding, W. 1948. A class of statistics with asymptotic normal distribution. *Annals of Mathematical Statistics*, 19: 293–325.

Ivanov, D.; and Ben-Porat, O. 2024. Personalized reinforcement learning with a budget of policies. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, 12735–12743.

Jayawardana, V.; Freydt, B.; Qu, A.; Hickert, C.; Yan, Z.; and Wu, C. 2025. IntersectionZoo: Eco-driving for Benchmarking Multi-Agent Contextual Reinforcement Learning. In *The Thirteenth International Conference on Learning Representations*.

Kang, Z.; Grauman, K.; and Sha, F. 2011. Learning with whom to share in multi-task feature learning. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, 521–528.

Lee, J.; Pacchiano, A.; Muthukumar, V.; Kong, W.; and Brunskill, E. 2021. Online model selection for reinforcement learning with function approximation. In *International Conference on Artificial Intelligence and Statistics*, 3340–3348. PMLR.

Mnih, V.; Kavukcuoglu, K.; Silver, D.; Rusu, A. A.; Veness, J.; Bellemare, M. G.; Graves, A.; Riedmiller, M. A.; Fidjeland, A. K.; Ostrovski, G.; Petersen, S.; Beattie, C.; Sadik, A.; Antonoglou, I.; King, H.; Kumaran, D.; Wierstra, D.; Legg, S.; and Hassabis, D. 2015. Human-level control through deep reinforcement learning. *Nature*, 518: 529–533.

Modi, A.; Jiang, N.; Singh, S.; and Tewari, A. 2018. Markov decision processes with continuous side information. In *Algorithmic Learning Theory*, 597–618. PMLR.

Schulman, J.; Wolski, F.; Dhariwal, P.; Radford, A.; and Klimov, O. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

Sobol', I. M. 1990. On sensitivity estimation for nonlinear mathematical models. *Matematicheskoe modelirovanie*, 2(1): 112–118.

Sodhani, S.; Zhang, A.; and Pineau, J. 2021. Multi-task reinforcement learning with context-based representations. In *International Conference on Machine Learning*, 9767–9779. PMLR.

Standley, T.; Zamir, A.; Chen, D.; Guibas, L.; Malik, J.; and Savarese, S. 2020. Which Tasks Should Be Learned Together in Multi-task Learning? In III, H. D.; and Singh, A., eds., *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, 9120–9132. PMLR.

Sun, L.; Zhang, H.; Xu, W.; and Tomizuka, M. 2022. Paco: Parameter-compositional multi-task reinforcement learning. *Advances in Neural Information Processing Systems*, 35: 21495–21507.

Teh, Y.; Bapst, V.; Czarnecki, W. M.; Quan, J.; Kirkpatrick, J.; Hadsell, R.; Heess, N.; and Pascanu, R. 2017. Distral: Robust multitask reinforcement learning. *Advances in neural information processing systems*, 30.

Tibshirani, R. J.; and Efron, B. 1993. An introduction to the bootstrap. *Monographs on statistics and applied probability*, 57(1): 1–436.

Turchetta, M.; Corinzia, L.; Sussex, S.; Burton, A.; Herrera, J.; Athanasiadis, I.; Buhmann, J. M.; and Krause, A. 2022. Learning long-term crop management strategies with cyclesgym. *Advances in neural information processing systems*, 35: 11396–11409.

Yagiura, M.; and Ibaraki, T. 2001. On Metaheuristic Algorithms for Combinatorial Optimization Problems. *Systems and Computers in Japan*, 32.

Yu, T.; Quillen, D.; He, Z.; Julian, R.; Hausman, K.; Finn, C.; and Levine, S. 2020. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. In *Conference on robot learning*, 1094–1100. PMLR.

Yücesoy, Y. E. Y.; and Tümer, M. B. 2015. Hierarchical Reinforcement Learning with Context Detection (HRL-CD). *International Journal of Machine Learning and Computing*, 5(5): 353–358.

Čech, E.; and Katětov, M. 1969. *Point Sets*. Academia, Publishing House of the Czechoslovak Academy of Sciences.