Pattern Transfer Learning for Reinforcement Learning in Order Dispatching

Runzhe Wan [1*], **Sheng Zhang**[1*], **Chengchun Shi**[2], **Shikai Luo**[3] and **Rui Song**[1]

[1] North Carolina State University

[2] London School of Economics and Political Science

[3] Tecent PCG

{rwan, szhang37}@ncsu.edu, c.shi7@lse.ac.uk, sluo198912@163.com, rsong@ncsu.edu

Abstract

Order dispatch is one of the central problems to ride-sharing platforms. Recently, value-based reinforcement learning algorithms have shown promising performance to solve this task. However, in real-world applications, the demand-supply system is typically nonstationary over time, posing challenges to re-utilizing data generated in different time periods to learn the value function. In this work, motivated by the fact that the relative relationship between the values of some states is largely stable across various environments, we propose a pattern transfer learning framework for value-based reinforcement learning in the order dispatch problem. Our method efficiently captures the value patterns by incorporating a concordance penalty. The superior performance of the proposed method is supported by experiments.

1 Introduction

One major task for large-scale ride-sourcing platforms, such as Uber and DiDi Chuxing, is to develop an order dispatch algorithm which matches order requests with idle drivers in real time. A high-quality dispatch algorithm can alleviate the traffic congestion problem, increase revenue for drivers, and serve customers better with higher answer rates [Xu *et al.*, 2018].

In recent years, value-based reinforcement learning (RL) algorithms have been widely used in the order dispatch problem [Tang *et al.*, 2019; Zhou *et al.*, 2021]. One major challenge to these algorithms is how to accurately estimate the value functions with limited real-time data. Although there are usually lots of logged historical data in ride-sourcing platforms, given the non-stationarity of the environment, the complex spatial-temporal dependency of this problem, and the multi-agent nature of this task, it is not clear how to re-utilize historical data generated by a different policy in a different time period, in a principled way [Qin *et al.*, 2020]. Naively combining all data sources may cause huge bias, while simply discarding historical data may cause large variance in value estimation and hence affect the dispatch quality. Most existing methods fail to address this important problem.

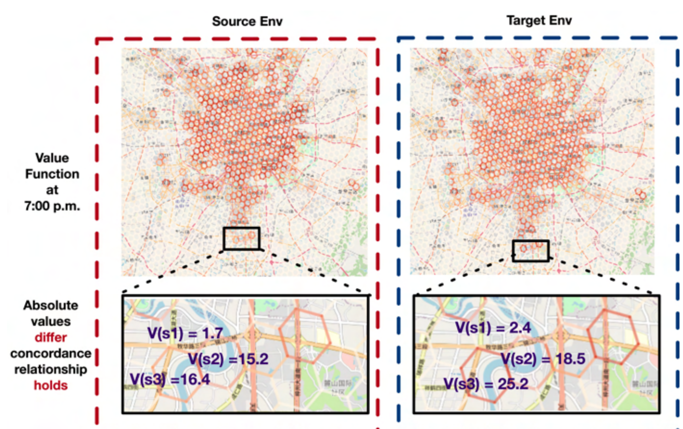

Figure 1: State values over two different time periods. Left and right panels denotes source and target environment, respectively. Computed with the data and setting introduced in Section 5 using Equation (2). A darker color indicates a higher value. The second row displays a zoom-in area, where it is clear that the absolute values in different time periods significantly differ while the relative relationship between states is consistent, which is referred to as the concordance relationship. Overall, the concordance relationship holds on more than 80% state pairs.

This paper is concerned with the following question: *how should we utilize data generated in a different environment to improve the efficiency of value function estimation in a new environment?* Such a goal may sound ambitious, and certain task-specific structures must be utilized. In this work, we focus on the concordance pattern, motivated by the following observations: in a demand-supply network, although the absolute value of a state may vary a lot across different environments, the relative relationship between some states tends to be stationary. For example, as shown in Figure 1, there exist some "hot zones" in the center of the town, which have higher values than the values of the "cold zones" in the suburbs. This relationship is stationary across different periods, and can be

*Equal contribution

Reinforcement Learning for Intelligent Transportation Systems (RL4ITS) Workshop in *the* 30^{th} *International Joint Conference on Artificial Intelligence (IJCAI-21)* , Montrealthemed Virtual Reality, Canada, 2021.

characterized by the concordance relationship between the value functions in different environments. Such a relationship also exists between, for example, the value of the rush hours and the normal hours. Therefore, such a pattern relationship can then be used to improve the efficiency of value estimation, by appropriately integrating a concordance penalty with a value-based RL algorithm.

In this paper, we investigate pattern transfer learning for the order dispatch problem. Our contributions are three-fold:

- Conceptually, we demonstrate a concrete approach to transfer the structural information learned from existing offline data to improve the efficiency of online value estimation. The high-level idea of utilizing a structural penalty function in transfer RL is generally applicable and is new to the literature.
- Methodologically, we develop a novel transfer RL algorithm for the order dispatch problem. The key ingredients of our method lay in appropriately constructing a concordance function and applying the concordance penalty to the value-based objective function.
- Empirically, we evaluate different algorithms in a well-calibrated simulator. The proposed method achieves superior performance and demonstrates the usefulness of pattern transfer.

2 Related Work

Order dispatching is a longstanding topic in the literature on intelligent transportation systems. Traditional methods typically only consider short-term performance. For example, Liao [2003] proposes to match the nearest driver to each order, Zhang and Pavone [2016] uses the queue data structure to dispatch with the first-come-first-serve strategy, and Zhang *et al.* [2017] aims to optimize the success rate of the order matches by matching driver-order pairs within a short time window.

However, these methods might lead to a sub-optimal policy in the long run, because they do not consider the long-term spatial equilibrium between the orders and drivers. Xu *et al.* [2018] made an important step forward by modeling the order dispatch problem as a Markov Decision Process (MDP) and designing a value-based RL algorithm, which can optimize resource allocation in a farsighted view and has shown great success in applications. Further extensions have been proposed in the literature. For example, Li et al. [Li *et al.*, 2019] extended the single-agent setting in Xu *et al.* [2018] to a multi-agent setting, which is more capable of modeling the complex interactions between drivers and orders. Tang et al. [Tang *et al.*, 2019] extended the tabular-like value function in Xu *et al.* [2018] to deep value networks. Our work builds on this line of research and considers the non-stationary nature of the order dispatch problem.

Transfer RL aims to boost the training process in a *target environment* by leveraging and transferring external knowledge from one or multiple *source environments* [Zhu *et al.*, 2020]. Existing transfer RL approaches can be roughly categorized into reward shaping [Williams and Baird, 1993; Devlin and Kudenko, 2012], learning from demonstrations [Bertsekas, 2011; Kim *et al.*, 2013], policy transfer [Rusu *et al.*, 2015; Czarnecki *et al.*, 2019], inter-task mapping [Gupta *et al.*, 2017; Torrey *et al.*, 2005], representation reuse [Rusu *et al.*, 2016], and learning disentangled representation [Dayan, 1993; Schaul *et al.*, 2015]. See Zhu *et al.* [2020] for a recent survey.

However, research on transfer RL in the order dispatch problem is limited. To the best of our knowledge, the only existing method is proposed by Wang *et al.* [2018], where several representation resue-type methods are adapted to transfer the pre-trained neural network model in the source environment to the target environment. Such an approach requires the value function to be modeled by a deep neural network so as to share the weights of hidden layers. Therefore, it is not directly applicable to other settings, such as the tabular setting as in [Xu *et al.*, 2018]. Our approach, on the contrary, is generally applicable to value function parameterized by any function class. In addition, our method is more interpretable in the transferring process. Specifically, in this work, we focus on the concordance relationship, which yields intuitive explanations and provides additional insights. Finally, our penalty-based approach can be easily extended when other kinds of domain knowledge about the connection between the source and the target environment are available, and it is of separate interest.

3 Setup

Order dispatch problem can be formalized as a semi-Markov decision process (SMDP) model [Sutton *et al.*, 1999], which is an extension of the MDP model in that SMDP allows the actions to be temporally extended. Various RL algorithms based on SMDP models have achieved great success in real-world order dispatch applications [Xu *et al.*, 2018; Wang *et al.*, 2018; Tang *et al.*, 2019; Qin *et al.*, 2020]. Formally, the components of the SMDP can be built as follows:

State. We consider an episodic setting with each day as one episode. The time and locations are discretized as T time points and N hexagon grids, respectively. The state space is $\mathcal{S} = \mathcal{T} \times G$, where $\mathcal{T} \equiv \{0, 1, \ldots, T\}$ and $G \equiv \{1, \ldots, N\}$. At each time $t \in \mathcal{T}$, the state of a driver is a temporal-spatial pair $s = (t, i)$, where $i \in G$ is the driver's location.

Action. An available driver can either be assigned to serve an order or to stay idle.

Reward. If one driver accepts an order with revenue R and the order takes time Δt, then we consider the reward $R_\gamma = \sum_{t=0}^{\Delta t} \gamma^t \frac{R}{\Delta t}$, where γ is the discount factor. If the driver chooses to stay idle, the reward is 0.

State transition. For an available driver at state $s = (t, i)$, if the action is to serve an order, the driver will transit to state $s' = (t + \Delta t, i')$, where Δt is the time cost and i' is the destination; otherwise, the driver will keep idling and transit to state $s' = (t + 1, i)$.

Value function. We use $V_\pi(s)$ to denote the expected discounted cumulative reward that one random driver can collect starting from state s to the end of the day, suppose the dispatch system follows the policy π. To simplify the notation, we may drop the subscript π, and use $V_{t,i}$ to denote $V((t, i))$.

Policy. The dispatching problem is a cooperative multi-agent RL task. At each decision point, the central agent will receive a list of order requests. A policy π will assign these orders to idle drivers. The goal is to learn an optimal policy that maximizes the expected long-term reward.

4 Pattern Transfer Learning in Order Dispatch

4.1 Value-based order dispatch

With such an SMDP model, we can design a generalized policy iteration (GPI) approach [Sutton and Barto, 2018] to optimize the long-term cumulative reward. A GPI framework alternates between a policy evaluation step where we evaluate the value of each state, and a policy improvement step where we behave greedily with respect to the value so as to improve the current policy. The framework is summarized in Algorithm 1, with two key components detailed below. We note that similar frameworks have been considered in the literature [Xu *et al.*, 2018; Tang *et al.*, 2019; Qin *et al.*, 2020], and the main differences lay in the details of the key components. In this work, a value transfer approach is designed to utilize existing offline data to evaluate the policy value more efficiently.

Algorithm 1 Generalized Policy Iteration for Order Dispatching

Data: transition buffer $\mathcal{D}$.
for day $1, 2, \ldots$ **do**
  Learn a value function $\widehat{V}$ from the data in $\mathcal{D}$ using a policy evaluation method (e.g., (6) or (2)).
  Within each dispatch window throughout the day, match orders and drivers in a collectively greedy way with respect to $\widehat{V}$ by solving (1).
  Add the new transition tuples into $\mathcal{D}$.
end for

Policy evaluation. At the beginning of each day, a data buffer of transitions tuples $\mathcal{D} = \{(s_j, a_j, r_j, s'_j)\}$ has been collected in previous days, where s_j is the initial state, a_j is the observed action, r_j is the received reward, and s'_j it the finish state. We need to evaluate the value of each state $V(s)$ using the collected data. Various methods have been proposed, such as dynamic programming [Xu *et al.*, 2018] and deep-Q network [Tang *et al.*, 2019]. We will detail our procedure in Section 4.3.

Online dispatch (policy improvement). Within each dispatch window, we need to match active orders and available drivers with the objective of maximizing the long-term collective cumulative rewards. We will act in a greedy way with respect to the estimated value function $\widehat{V}$. Specifically, as a common procedure in the literature [Xu *et al.*, 2018; Tang *et al.*, 2019], we consider a bipartite matching problem for every possible driver-order pair:

$$\arg\max_{\{a_{lk}\}\in C} \sum_{l=0}^{m} \sum_{k=0}^{n} \widehat{Q}(l,k) a_{lk}, \tag{1}$$

where $l \in \{1, \ldots, m\}$ corresponds to all available drivers, $k \in \{1, \ldots, n\}$ corresponds to the active orders, and $a_{lk} \in \{0, 1\}$ is the indicator of assigning order k to driver l with $l = 0$ or $k = 0$ denoting no match. Here, C contains constraints including (i) $\sum_{l=0}^{m} a_{lk} = 1, \forall k$, indicating that each order can be assigned to at most one driver, (ii) $\sum_{k=0}^{n} a_{lk} = 1, \forall l$, meaning that each driver can take to at most one order, and (iii) some other business constraints. The Q-function can be derived as $\widehat{Q}(l,k) = \gamma^{\Delta t(l,k)} \widehat{V}_{i'_k, t'_{lk}} + r_k$, where $\Delta t(l,k)$ is the time cost, i'_k is the finish location, t'_{lk} is the finish time, r_k is the reward of order k. It is easy to verify that $\widehat{Q}(l,0) = \widehat{V}(s_l)$, where s_l is the current state of driver l. The Kuhn-Munkres (KM) algorithm [Munkres, 1957] can be applied to solve (1), and the advantage function trick in Xu *et al.* [2018] can be used to reduce the computational cost.

4.2 Concordance relationship

As discussed in Section 1, we aim to utilize the pattern similarity between the old (source) environment and the current (target) environment through a concordance relationship between their value functions. More precisely, suppose we have a value function V^s learned by previous interactions with an source SMDP environment $\mathcal{M}^s$. V^s could be learned by running Algorithm 1 or other RL algorithms in this environment, or estimated from logged data. Our goal is to run Algorithm 1 in the target environment $\mathcal{M}^t$ with the objective of maximizing the cumulative rewards. We assume $\mathcal{M}^t$ and $\mathcal{M}^s$ share the same state space, rewarding system, and discount factor. However, due to environment non-stationarity, the spatial-temporal distribution of orders and drivers may change significantly, and so does the value function. Therefore, directly using the old dataset or value functions may cause huge bias.

Motivated by the observations in Figure 1 that the relative relationship between the value of the hot regions and of the cold regions (or that between the rush hours and the normal hours) will be relatively consistent, we aim to capture this structural stability so as to transfer knowledge from the old data to stabilize the value estimation. Formally, the *concordance relationship* on a state pair (s_1, s_2) holds between two value functions V and V' if and only if

$$[V(s_1) - V(s_2)][V'(s_1) - V'(s_2)] \geq 0.$$

Given some pre-specified or estimated distribution μ over the space of state pairs $\mathcal{S} \times \mathcal{S}$, we define the *concordance loss* between two value functions V and V' as

$$c(V, V'; \mu) \equiv \mathbb{E}_{(s_1,s_2)\sim\mu} \mathbb{I}\big\{[V(s_1) - V(s_2)] \times [V'(s_1) - V'(s_2)] < 0\big\},$$

where $\mathbb{I}(\cdot)$ denotes indicator function. Here, $c(V, V'; \mu)$ is the probability that the concordance relationship between V and V' will be violated, evaluated on μ. The concordance function has been widely employed in applications such as classification tasks [Cortes and Vapnik, 1995] and optimal decision making [Liang *et al.*, 2017; Fan *et al.*, 2017; Shi *et al.*, 2021]. However, to the best of our knowledge, it is used in RL for the first time.

Let the optimal value function in the target environment be V^*. Motivated by the discussions above, we make the following assumption throughout this paper: the concordance relationship between V^* and V^s will hold with high probability, as evaluated on some appropriately chosen distribution μ. More precisely, $c(V^*, V^s; \mu) \in [0, 1)$ is small.

In practice, for each state pair (s_1, s_2), $\mu(s_1, s_2)$ should incorporate important domain knowledge about the importance of this pair as well as our belief that the concordance relationship on this pair will hold between V^* and V^s. As an example, in this paper, we focus on the value concordance relationship between hot regions and cold regions. Specifically, let E be a set of user-specified location pairs on which the concordance relationship is believed to hold, we define

$$\mu(s_1, s_2) = (T|E|)^{-1}\mathbb{I}[(g(s_1), g(s_2)) \in E, t(s_1) = t(s_2)],$$

where $|E|$ is the cardinality of E, and $g(s)$ and $t(s)$ is the location and time component of the state s, respectively.

4.3 Policy evaluation with concordance penalty

In this section, we discuss how to improve the efficiency of value function estimation by utilizing the dataset collected in the source environment through a concordance penalty function. To simplify the notation, for every t, we denote $\boldsymbol{V}_t = \{V_{t,i}\}_{i=1}^N$. We similarly define $\boldsymbol{V}_t^s$ and $\widehat{\boldsymbol{V}}_t$.

A straightforward approach to estimate the value function is dynamic programming (DP) [Xu *et al.*, 2018]. Let $\widehat{V}_{T,i} = 0$ for every $i \in G$. For $t = T-1, T-2, \ldots, 0$, $\widehat{V}_{t,i}$ for every $i \in G$ is calculated as

$$\widehat{V}_{t,i} = \frac{1}{|\mathcal{D}(t,i)|} \sum_{j \in \mathcal{D}(t,i)} (\gamma^{\Delta t(a_j)} \widehat{V}_{t'_j, i'_j} + r_j), \tag{2}$$

where $s'_j = (i'_j, t'_j)$ and $\mathcal{D}(t,i) = \{j : s_j = (t,i)\}$ denoting tuples with current state (t, i).

To present our method, we note that the DP-based policy evaluation step (2) is equivalent to minimizing the squared temporal-difference (TD) error [Sutton and Barto, 2018]. Specifically, we first set $\widehat{V}_{t,i} = 0$ for every i, and then solve the following optimization problem recursively, for $t = T-1, T-2, \ldots, 0$:

$$\widehat{\boldsymbol{V}}_t = \arg\min_{\boldsymbol{V}_t} \sum_{j \in \mathcal{D}(t)} [V_{t,i_j} - \gamma^{\Delta t(a_j)} \widehat{V}_{t'_j, i'_j} - R_\gamma(a_j)]^2. \tag{3}$$

It is easy to verify that, the estimated value function $\widehat{\boldsymbol{V}}$ by solving (3) is the same with the output of (2).

With such an observation, we propose to estimate the value function by minimizing the squared TD error with the concordance constraint. For any time index t and any two value functions V and V', we define the *spatial concordance loss* between $\{V_{t,i}\}_{i=1}^N$ and $\{V'_{t,i}\}_{i=1}^N$ as

$$l(\{V_{t,i}\}_{i=1}^N, \{V'_{t,i}\}_{i=1}^N; E) \equiv \sum_{(i,j) \in E} \mathbb{I}\Big\{[V_{t,i} - V_{t,j}] \times [V'_{t,i} - V'_{t,j}] < 0\Big\}.$$

Then, let $\mathcal{D}(t) = \bigcup_i \mathcal{D}(t,i)$, we can obtain $\widehat{\boldsymbol{V}}_t$ by solving

$$\begin{aligned} &\arg\min_{\boldsymbol{V}_t} \sum_{j \in \mathcal{D}(t)} \Big[V_{t,i_j} - \gamma^{\Delta t(a_j)} \widehat{V}_{t'_j, i'_j} - R_\gamma(a_j)\Big]^2 \\ &s.t. \qquad l(\boldsymbol{V}_t, \boldsymbol{V}_t^s; E) \le \epsilon. \end{aligned} \tag{4}$$

To solve this constrained optimization problem, a equivalent penalized optimization problem is considered:

$$\begin{aligned} \arg\min_{\boldsymbol{V}_t} \Big\{ &\sum_{j \in \mathcal{D}(t)} \Big[V_{t,i_j} - \gamma^{\Delta t(a_j)} \widehat{V}_{t'_j, i'_j} - R_\gamma(a_j)\Big]^2 \\ &+ \lambda \times l(\boldsymbol{V}_t, \boldsymbol{V}_t^s; E)\Big\}, \end{aligned} \tag{5}$$

where $\lambda > 0$ is the Lagrange parameter. By the Lagrange duality, we know that, for any $\epsilon > 0$, there exists some $\lambda > 0$ such that the solution of (5) is the same with that of (4).

Finally, we note that problem (5) is not differentiable. In practice, the hinge loss, which is a convex upper bound of the concordance loss function, has been commonly used as a surrogate loss function [Liang *et al.*, 2017; Cortes and Vapnik, 1995]. Specifically, the hinge loss between $\boldsymbol{V}_t$ and $\boldsymbol{V}_t^s$, can be written as

$$\begin{aligned} h(\boldsymbol{V}_t, \boldsymbol{V}_t^s; E) = \sum_{(i,j) \in E} &\Big\{\mathbb{I}[V_{t,i}^s < V_{t,j}^s][1 - (V_{t,j} - V_{t,i})]_+ \\ &+ \mathbb{I}[V_{t,i}^s > V_{t,j}^s][1 - (V_{t,i} - V_{t,j})]_+\Big\} \end{aligned}$$

Putting all the discussions together, we propose to replace the DP-based policy evaluation step (2) by solving the following optimization problem recursively, for $t = T-1, \ldots, 0$:

$$\begin{aligned} \widehat{\boldsymbol{V}}_t = \arg\min_{\boldsymbol{V}_t} \Big\{ &\sum_{j \in \mathcal{D}(t)} \Big[V_{t,i_j} - \gamma^{\Delta t(a_j)} \widehat{V}_{t'_j, i'_j} - R_\gamma(a_j)\Big]^2 \\ &+ \lambda \times h(\boldsymbol{V}_t, \boldsymbol{V}_t^s; E)\Big\}. \end{aligned} \tag{6}$$

Optimization. Let the objective function of (6) be $\mathcal{L}(\boldsymbol{V}_t; \boldsymbol{V}_t^s, \mathcal{D}(t), \lambda)$. For every $i = 1, \ldots, N$, we can derive the partial gradient as

$$\begin{aligned} &\frac{\partial}{\partial V_{i,t}} \mathcal{L}(\boldsymbol{V}_t; \boldsymbol{V}_t^s, \mathcal{D}(t), \lambda) \\ &= 2 \sum_{j \in \mathcal{D}(i,t)} (V_{t,i_j} - \gamma^{\Delta t(a_j)} \widehat{V}_{t'_j, i'_j} - R_\gamma(a_j)) \\ &- \lambda \sum_{j:(i,j) \in E} \big[\mathbb{I}(V_{t,i}^s < V_{t,j}^s, V_{t,j} - V_{t,i} < 1) \\ &+ \mathbb{I}(V_{t,i}^s > V_{t,j}^s, V_{t,i} - V_{t,j} < 1).\big] \end{aligned} \tag{7}$$

The explicit form of the gradient $\frac{\partial}{\partial \boldsymbol{V}_t} \mathcal{L}(\boldsymbol{V}_t; \boldsymbol{V}_t^s, \mathcal{D}(t), \lambda)$ then follows. To solve (6), we apply gradient descent with step sizes chosen by a diminishing step size rule [Boyd *et al.*, 2003]. We have the following convergence guarantee.

Proposition 1 (Convergence) *With a diminishing step size rule, our gradient descent optimization algorithm will converge to the solution of* (6).

Proof. For the objective function, both the loss part and the penalty part is a composition of a convex function and an affine function, and hence it is convex. Besides, because the hinge loss is a subdifferentiable function, it is easy to verify that the objective function is also a subdifferentiable function. Therefore, according to Boyd *et al.* [2003], a gradient descent optimization algorithm with a diminishing step size schedule for a convex subdifferentiable objective function will converge to the global optimum.

5 Experiments

Simulator. To evaluate the proposed method, we build a real data calibrated dispatch simulator. Such simulator are constructed based on the open dataset from the DiDi ridesharing platform . This dataset contains drivers' trajectories, transition probability of idle drivers, information of order requests and hexagonized map grids in Chengdu, China, for 30 days. Our simulator design follows the procedures introduced in Xu *et al.* [2018]. Specifically, the order requests and drivers' online time periods are kept the same with the real data. After logging-in, the drivers will completely follow the dispatch algorithm. Other information such as the transition probability of idle drivers and the cancellation rates are all provided by or fitted from the data. The difference between the simulated results from our simulator and the official simulator is less than 5% in terms of answer rate and total GMV.

Setting. Following Xu *et al.* [2018], we use the first 15 days as the source environment and the latter 15 days as the target environment. In the real dataset, there exists a huge difference in the environment between weekdays and weekends. In this experiment, we focus on the weekdays only. Different value-based dispatch policies are run during 11 weekdays in the latter half a month and their performance is recorded. All of these policies use the KM algorithm (1) for dispatch, and the only difference lies in the choice of the Q-values. For policies relying on data generated in the target environment, we use the greedy policy as the initial policy on the first day. The following policies are considered:

- **Greedy (Myopia)**: Only instant order rewards are considered. Replace $\widehat{Q}(l,k)$ in (1) by R_{lk}.
- **Source-only**: The value functions are calculated using the source data only. Details can be found in Xu *et al.* [2018].
- **Target-only**: The value function is initialized with zero and updated for the latter 15 days using TD updates (Equation 3) with no penalty.
- **Naively-combine**: The value functions are calculated from the first 15 days and updated for the latter 15 days using TD updates with no penalty.
- **Pattern-transfer**: Our proposed concordance penalty-based value evaluation methods.

Results. The performance of different methods is summarized in Figure 2. We consider two choices of γ, 0.9 and 0.95. In addition to the cumulative rewards, we also report the answer rates (proportion of orders being answered) and the completion rates (proportion of accepted orders being eventually completed). Both metrics are commonly used to reflect the quality of a dispatch algorithm [Xu *et al.*, 2018]. We summarize our findings as follows. We focus on the cumulative reward as the main evaluation metric.

- The proposed method outperforms the baselines by a significant margin. Specifically, compared with the target-only algorithm (i.e., no pattern transfer), the proposed method enjoys a jumpstart in the first several days, converges more quickly, and the convergence performance is better. These improvements come from the efficient pattern transfer via the concordance penalty;
- The performance of the target-only algorithm also improves slowly as the data accumulates, but the rate of improvement is much slower than the proposed method;
- Both the source-only and the naively-combine algorithms suffer from the bias incurred by the non-stationarity of the environment;
- The greedy policy does not consider the long-term performance, will cause undesired demand-supply distribution, and hence performs the worst.

To see the convergence speed of different methods under the GPI framework more clearly, we conduct an extra experiment, where we repeat the simulation of a single day multiple times. Results are shown in Figure 3. The performance of our method typically converges within 2 iterations and achieves superior performance, while the value of the target-only algorithm increases at a much slower rate. This experiment further demonstrates the usefulness of the proposed pattern transfer method.

6 Discussion

In this paper, we propose a novel pattern transfer method for the online order-dispatch problem. At the heart of our method is a concordance penalty, which efficiently captures the value patterns. Integrated with the GPI framework, the algorithm demonstrates superior performance in dealing with non-stationary environments.

There are several future directions worthy of study. First, we currently model the value function without function approximation. It would be interesting to couple our proposal with some state-of-the-art universal function approximators, e.g., deep neural networks. Second, we can consider applying the pattern transfer learning method to more complex problems in intelligent transportation systems, such as multi-agent RL for order dispatch, joint order dispatching and fleet management, etc. Third, the concordance relationship is only one kind of pattern, and the idea of penalty-based transfer RL can be more general than the setup considered in this paper. it is practically interesting to apply the proposed methodology to other domains to evaluate its empirical performance. Lastly, providing theoretical guarantees for the proposed method is also a meaningful next step.

https://gaia.didichuxing.com

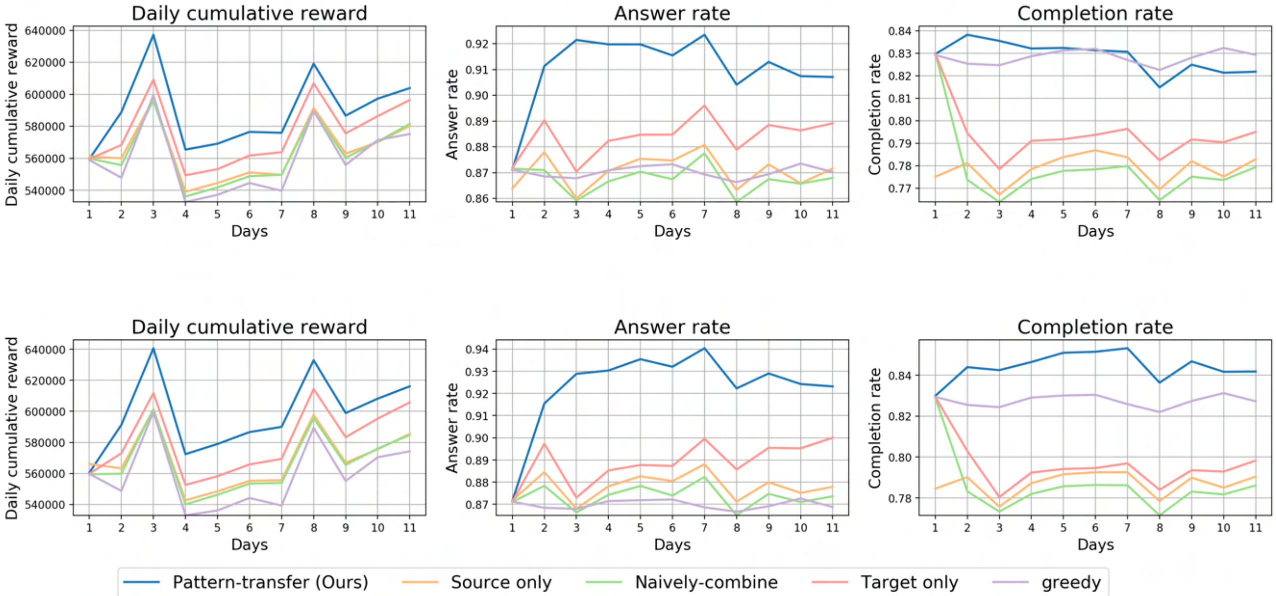

Figure 2: Performance of different methods when $\gamma = 0.9$ (upper) and $\gamma = 0.95$ (lower). The x-axis represents consecutive weekdays in the target environment. Our method outperforms the baseline methods under different metrics.

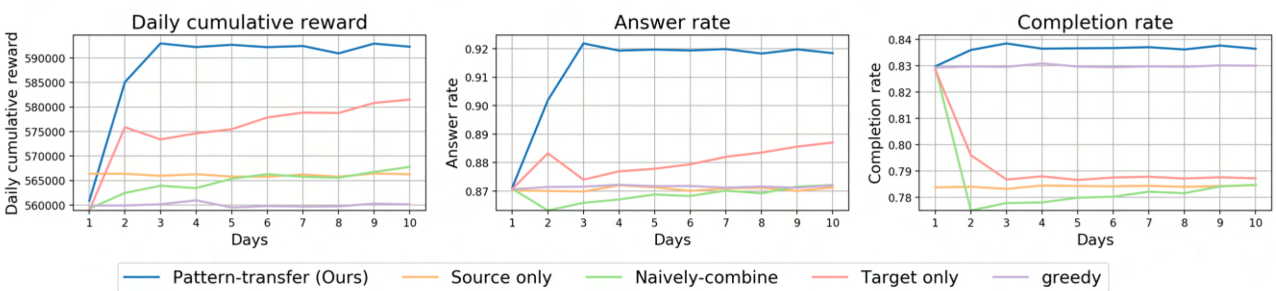

Figure 3: Results for different methods when the same day is repeatedly simulated for multiple times. The x-axis represents repeated iterations of this single day in the target environment. Our method shows a stable performance within 2 iterations, while the target-only method requires more iterations to improve and converge.

Acknowledgement

We thank Didi Chuxing for sharing the datasets for public research.

References

Dimitri P Bertsekas. Approximate policy iteration: A survey and some new methods. *Journal of Control Theory and Applications*, 9(3):310–335, 2011.

Stephen Boyd, Lin Xiao, and Almir Mutapcic. Subgradient methods. *lecture notes of EE392o, Stanford University, Autumn Quarter*, 2004:2004–2005, 2003.

Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

Wojciech M Czarnecki, Razvan Pascanu, Simon Osindero, Siddhant Jayakumar, Grzegorz Swirszcz, and Max Jaderberg. Distilling policy distillation. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 1331–1340. PMLR, 2019.

Peter Dayan. Improving generalization for temporal difference learning: The successor representation. *Neural Computation*, 5(4):613–624, 1993.

Sam Michael Devlin and Daniel Kudenko. Dynamic potential-based reward shaping. In *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems*, pages 433–440. IFAAMAS, 2012.

Caiyun Fan, Wenbin Lu, Rui Song, and Yong Zhou. Concordance-assisted learning for estimating optimal individualized treatment regimes. *Journal of the Royal Statistical Society. Series B, Statistical methodology*, 79(5):1565, 2017.

Abhishek Gupta, Coline Devin, YuXuan Liu, Pieter Abbeel, and Sergey Levine. Learning invariant feature spaces to transfer skills with reinforcement learning. *arXiv preprint arXiv:1703.02949*, 2017.

Beomjoon Kim, Amir-massoud Farahmand, Joelle Pineau, and Doina Precup. Learning from limited demonstrations. In *Advances in Neural Information Processing Systems*, pages 2859–2867, 2013.

Minne Li, Zhiwei Qin, Yan Jiao, Yaodong Yang, Jun Wang, Chenxi Wang, Guobin Wu, and Jieping Ye. Efficient ridesharing order dispatching with mean field multi-agent reinforcement learning. In *The World Wide Web Conference*, pages 983–994, 2019.

Shuhan Liang, Wenbin Lu, Rui Song, and Lan Wang. Sparse concordance-assisted learning for optimal treatment decision. *J. Mach. Learn. Res.*, 18:202–1, 2017.

Ziqi Liao. Real-time taxi dispatching using global positioning systems. *Communications of the ACM*, 46(5):81–83, 2003.

James Munkres. Algorithms for the assignment and transportation problems. *Journal of the society for industrial and applied mathematics*, 5(1):32–38, 1957.

Zhiwei Qin, Xiaocheng Tang, Yan Jiao, Fan Zhang, Zhe Xu, Hongtu Zhu, and Jieping Ye. Ride-hailing order dispatching at didi via reinforcement learning. *INFORMS Journal on Applied Analytics*, 50(5):272–286, 2020.

Andrei A Rusu, Sergio Gomez Colmenarejo, Caglar Gulcehre, Guillaume Desjardins, James Kirkpatrick, Razvan Pascanu, Volodymyr Mnih, Koray Kavukcuoglu, and Raia Hadsell. Policy distillation. *arXiv preprint arXiv:1511.06295*, 2015.

Andrei A Rusu, Neil C Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. Progressive neural networks. *arXiv preprint arXiv:1606.04671*, 2016.

Tom Schaul, Daniel Horgan, Karol Gregor, and David Silver. Universal value function approximators. In *International conference on machine learning*, pages 1312–1320. PMLR, 2015.

Chengchun Shi, Rui Song, Wenbin Lu, et al. Concordance and value information criteria for optimal treatment decision. *Annals of Statistics*, 49(1):49–75, 2021.

Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

Richard S Sutton, Doina Precup, and Satinder Singh. Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning. *Artificial intelligence*, 112(1-2):181–211, 1999.

Xiaocheng Tang, Zhiwei Qin, Fan Zhang, Zhaodong Wang, Zhe Xu, Yintai Ma, Hongtu Zhu, and Jieping Ye. A deep value-network based approach for multi-driver order dispatching. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 1780–1790, 2019.

Lisa Torrey, Trevor Walker, Jude Shavlik, and Richard Maclin. Using advice to transfer knowledge acquired in one reinforcement learning task to another. In *European Conference on Machine Learning*, pages 412–424. Springer, 2005.

Zhaodong Wang, Zhiwei Qin, Xiaocheng Tang, Jieping Ye, and Hongtu Zhu. Deep reinforcement learning with knowledge transfer for online rides order dispatching. In *2018 IEEE International Conference on Data Mining (ICDM)*, pages 617–626. IEEE, 2018.

Ronald J Williams and Leemon C Baird. Tight performance bounds on greedy policies based on imperfect value functions. Technical report, Citeseer, 1993.

Zhe Xu, Zhixin Li, Qingwen Guan, Dingshui Zhang, Qiang Li, Junxiao Nan, Chunyang Liu, Wei Bian, and Jieping Ye. Large-scale order dispatch in on-demand ride-hailing platforms: A learning and planning approach. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 905–913. ACM, 2018.

Rick Zhang and Marco Pavone. Control of robotic mobility-on-demand systems: a queueing-theoretical perspective. *The International Journal of Robotics Research*, 35(1-3):186–203, 2016.

Lingyu Zhang, Tao Hu, Yue Min, Guobin Wu, Junying Zhang, Pengcheng Feng, Pinghua Gong, and Jieping Ye. A taxi order dispatch model based on combinatorial optimization. In *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 2151–2159, 2017.

Fan Zhou, Shikai Luo, Xiaohu Qie, Jieping Ye, and Hongtu Zhu. Graph-based equilibrium metrics for dynamic supply-demand systems with applications to ride-sourcing platforms. *arXiv preprint arXiv:2102.05805*, 2021.

Zhuangdi Zhu, Kaixiang Lin, and Jiayu Zhou. Transfer learning in deep reinforcement learning: A survey. *arXiv preprint arXiv:2009.07888*, 2020.