Domain Knowledge Guided Offline Q Learning

Xiaoxuan Zhang* **Sijia Zhang*** **Yen-Yun Yu**

Ancestry.com Operations Inc.

Abstract

Offline reinforcement learning (RL) is a promising method for applications where direct exploration is not possible but a decent initial model is expected for the online stage. In practice, offline RL can underperform because of overestimation attributed to distributional shift between the training data and the learned policy. A common approach to mitigating this issue is to constrain the learned policies so that they remain close to the fixed batch of interactions. This method is typically used without considering the application context. However, domain knowledge is available in many real-world cases and may be utilized to effectively handle the issue of out-of-distribution actions. Incorporating domain knowledge in training avoids additional function approximation to estimate the behavior policy and results in easy-to-interpret policies. To encourage the adoption of offline RL in practical applications, we propose the Domain Knowledge guided Q learning (DKQ). We show that DKQ is a conservative approach, where the unique fixed point still exists and is upper bounded by the standard optimal Q function. DKQ also leads to lower chance of overestimation. In addition, we demonstrate the benefit of DKQ empirically via a novel, real-world case study - guided family tree building, which appears to be the first application of offline RL in genealogy. The results show that guided by proper domain knowledge, DKQ can achieve similar offline performance as standard Q learning and is better aligned with the behavior policy revealed from the data, indicating a lower risk of overestimation on unseen actions. Further, we demonstrate the efficiency and flexibility of DKQ with a classical control problem.

1 Introduction

In recent years, reinforcement learning (RL) has been increasingly applied in a variety of domains as an elegant approach for sequential decision making and data-driven personalization [1, 2]. Offline RL with a fixed batch of historical trajectories is a data-efficient strategy for applications where further interactions with the environment is expensive or unfeasible. Even if online reinforcement is achievable, we can still utilize offline RL to provide a good initial policy. However, the adoption of offline RL can be challenging in practice due to several limitations, such as distributional shift between the new policy and the batch data [3, 4], unrealistic reward estimation of unseen actions [5, 6], and challenges in approximating the unknown behavior policy from training trajectories [7, 8]. Some of these barriers may be overcome by considering the application context and available domain knowledge. For example, if we have domain knowledge that contains information about the action distribution in behavioral data, we can use it to sample actions to reduce distributional shift in training while avoiding errors introduced by generating behavioral models. The main goal of this paper is to offer a framework, Domain Knowledge guided Q learning (DKQ), that makes the integration of domain knowledge and offline RL possible and flexible. To further bridge the gap between offline RL and real-world applications, in addition to theoretically proving the validity of the proposed DKQ method, we showcase its usage and benefits with a unique industry application – guided family tree

*Correspondence: xzhang@ancestry.com, sizhang@ancestry.com

Offline Reinforcement Learning Workshop at Neural Information Processing Systems, 2021

building for a popular online genealogy service. To the best of our knowledge, this is the first study that applies RL to provide data-driven and metric-driven guidance for family tree building.

Genealogy-oriented businesses harness the information in historical records to help customers trace ancestry and build family trees. Family trees represent family members as nodes and demonstrate family relationships (father, spouse, etc.) and degrees of relationship (kinship distance). Recent development in internet technologies and artificial intelligence have allowed digitization of billions of historical records and encouraged millions of people to build and share their family trees online [9–11]. However, many of the genealogy website users are amateur. Unlike professional genealogist, amateur users may not have the knowledge to design an effective research plan. This can lead to dead ends on family trees or difficulty in trawling through a large amount of historical records available to users. One possible approach to help amateur users is to design a tree building policy that suggests the next best node to work on. We formulate Next Best Node as an RL problem since tree building is a sequential task. DKQ can provide a suitable solution because it is capable of learning good policies from a diverse set of tree building trajectories while incorporating domain knowledge from genealogists or previous data analysis. An offline RL setting reduces potential disturbances of customer experience and ensures performance before deployment.

DKQ calculates the weighted average of Q value estimates from multiple action subsets, which can be selected based on domain knowledge. We show through theoretical analysis that DKQ can result in conservative Q values. The case study demonstrates that DKQ reduces the chance of obtaining unrealistic estimates for unseen actions, a common problem in offline RL [5, 6]. Besides genealogy, this method has the potential to be extended to other products and services where domain knowledge and rich user data are readily available. We summarize our major contributions as follows:

- We propose a framework named Domain Knowledge guided Q learning (DKQ) that flexibly incorporates domain knowledge without introducing additional function approximation.
- We theoretically study the properties of DKQ in the tabular setting by proving that DKQ is a conservative approach and the sub-optimality is bounded as a function of domain knowledge.
- We illustrate the benefit of DKQ via a real-world case study, which, to the best of our knowledge, is the first study that applies RL to provide guidance for family tree building.

2 Related Work

Offline RL is a promising learning approach when online interactions are costly or unfeasible [12–14]. For example, in healthcare applications, avoiding online policy updates mitigates safety concerns. For recommender systems, extensive offline evaluation helps ensure positive customer experience. Despite the benefits of data-driven RL in real-world applications, effective learning with fixed offline batch is difficult to achieve. A major challenge is extrapolation error, defined as unseen action-state pairs getting unrealistic Q-value estimates [5, 3]. Such error can be propagated during training when greedy exploitation is employed to select actions, leading to suboptimal policies.

To correct for extrapolation error and address the underlying distributional mismatch between the new policy and the batch data, two categories of solutions have been developed: (1) restricting the policy to lie close to the behavior policy and (2) correcting for overly loose Q estimates by penalizing uncertainty [4]. A few recent algorithms are particularly relevant to our work. The Batch-Constrained deep Q-learning (BCQ) [5] considers only candidate actions sampled from a perturbed generative model in order to strike a balance between staying close to the batch and increasing the diversity of actions. Further, a modified Clipped Double Q-learning approach [15] is used to penalize rare or unseen states. Another method to control for uncertainly is bootstrapping ensembles of Q functions. For instance, the Random Ensemble Mixture model [16] demonstrates that even simple random convex combination of various Q-value estimates can enhance the generalization in offline RL.

Although constraining actions and averaging Q-values with random weights performed well on common benchmarks, they did not consider the application context or domain knowledge. A recent study [7] following BCQ suggests the importance of generative model design for estimating behavior policies in offline RL. If we already have domain knowledge for a specific application, we could directly incorporate domain knowledge in policy learning and thus avoid errors from estimating behavior policies. Prior work shows that domain knowledge can be used to warm start reinforcement learning and tighten regret bound [17, 18]. An Expert-Supervised RL framework has been developed to allow for application-specific risk averse implementation to learn safe and optimal policies in

domains like healthcare [8]. However, the training data needs to be collected from experts to impose safety constraints, and it cannot readily handle continuous state space or cases where not all actions are always available. Given limitations of existing approaches, we propose DKQ to incorporate domain knowledge and various action conditions from a diverse set of trajectories.

Fitted Q-iteration (FQI) is a class of offline RL that exploits regression algorithms to approximate the Q function by iteratively expanding the optimization horizon [19]. It has become increasingly popular due to good performance and data efficiency [4, 20]. However, FQI may still fail to find the optimal policy because of extrapolation error, especially when teh data is high-dimensional[21, 4]. Although convergence is not guaranteed, FQI will not lead to unbounded divergence [19, 21]. We prove that DKQ with FQI as the function appoximator preserves this property.

3 Preliminaries

Reinforcement learning is typically described as a Markov Decision Process (MDP) defined by a tuple $(\mathcal{S}, \mathcal{A}, r, P, \gamma)$. $\mathcal{S}$ and $\mathcal{A}$ respectively represent the state and action space. $P(\mathbf{s}'|\mathbf{s}, \mathbf{a})$ represents the dynamics (transition probability), and $r(\mathbf{s}, \mathbf{a})$ is the reward function. We assume that r is uniformly bounded by R_l and R_u, i.e., $R_l \leq r(\mathbf{s}, \mathbf{a}) \leq R_u$, $\forall \mathbf{s}, \mathbf{a}$. $\gamma \in [0, 1)$ is the discount factor. The goal is to learn a policy $\pi(\mathbf{a}|\mathbf{s})$ that maximizes the expected discounted cumulative reward. The action-value function for policy π is defined as $Q_\pi(\mathbf{s}, \mathbf{a}) = \mathbb{E}_\pi[\sum_{t=0}^{\infty} \gamma^t r(\mathbf{s}_t, \mathbf{a}_t)|S_0 = \mathbf{s}, A_0 = \mathbf{a}]$. The optimal action value function is $Q^*(\mathbf{s}, \mathbf{a}) = \max_\pi Q_\pi(\mathbf{s}, \mathbf{a})$. Let $\mathcal{T}$ represent the Bellman optimality operator for action-value function defined as (1), then Q^* is the fixed point of $\mathcal{T}$.

$$\mathcal{T}Q(\mathbf{s}, \mathbf{a}) = r(\mathbf{s}, \mathbf{a}) + \gamma \mathbb{E}_{\mathbf{s}' \sim P(\mathbf{s}'|\mathbf{s}, \mathbf{a})} \big[\max_{\mathbf{a}'} Q(\mathbf{s}', \mathbf{a}') \big]. \tag{1}$$

Existing research of offline RL, such as [5–7], mainly focuses on resolving the issue of inaccurate value estimates of out-of-distribution actions. Those studies assume no knowledge of the underlying domain and estimate the unknown behavior policy from data. However, in many real-world applications, like the case studied in this paper, domain knowledge is indeed available. In those cases, incorporating the domain knowledge into training procedures could be more effective and valuable than simply trying to mitigate the problem of distributional shift without the application context.

Definition 1. *We define domain knowledge as $\mathcal{F} = \{(f_i, \alpha_i)\}_{i=1}^K$, where $f_i : \mathcal{A} \to \{0, 1\}$ is an indicator function that describes a type of actions, and $\alpha_i \in [0, 1]$ is a normalized non-negative weight of importance, i.e., $\sum_{i=1}^K \alpha_i = 1$.*

The domain knowledge $\mathcal{F}$ is interpreted as follows: If an action meets the constraint defined by f_i, it has the importance measured by α_i. An example of the indicator function is $f(\mathbf{a}) = \mathbb{I}_{[\mathbf{e}_j^\top \mathbf{a} > 0]}$, which describes the action set $\{\mathbf{a} \in \mathcal{A}|\mathbf{a}[j] > 0\}$, where $\mathbf{e}_j$ is the unit vector on the direction of j-th axis and $\mathbf{a}[j]$ is the j-th element of $\mathbf{a}$. If domain experts believe that certain types of actions are no recommended, their weights can be set to zero. Besides, we denote the data set as $\mathcal{D} = \{(\mathbf{s}_t, \mathbf{a}_t, r_t, \mathbf{s}'_t, \tau_t)\}_{t=0}^T$, where τ_t is an indicator of $termination$.

4 Domain Knowledge Guided Q Learning

In industry applications, domain knowledge is typically available. Companies can gain insights of their products and customers from various channels, such as A/B tests, customer research and domain experts. Offline RL is a promising approach when online interactions are expensive or disturb customer experience. Even if companies plan to reinforce the model in an online setting, a well-performing initial policy trained offline with expert knowledge would still be beneficial. Incorporating domain knowledge into training has the potential to result in better policies because we make the learning process both data driven and human knowledge guided. However, domain knowledge can be too complicated or obscure to be formulated as a policy, which is expressed as a conditional probability distribution. To address this issue, we first provide a flexible definition of domain knowledge (Definition 1), then propose the Domain Knowledge guided Q Learning (DKQ).

4.1 Q Operator Guided by Domain Knowledge

By convention, we provide theoretical analysis of the properties for tabular MDP setting. As a result of page limit, all missing proofs are placed in the Appendix.

We define the Q operator guided by domain knowledge $\mathcal{F}$ as $\mathcal{T}_{\mathcal{F}}$. Specifically,

$$\mathcal{T}_{\mathcal{F}}Q(\mathbf{s},\mathbf{a}) \coloneqq r(\mathbf{s},\mathbf{a}) + \gamma \mathbb{E}_{\mathbf{s}'\sim P(\mathbf{s}'|\mathbf{s},\mathbf{a})}\big[\sum_{i=1}^{K} \alpha_i \max_{\mathbf{a}'\in supp(f_i)} Q(\mathbf{s}',\mathbf{a}')\big], \tag{2}$$

where $supp(f_i)$ represents the support of f_i. In the case where $K = 1$ (leading to $\alpha_1 = 1$) and $supp(f_1) = \mathcal{A}$, $\mathcal{T}_{\mathcal{F}}$ becomes the Bellman optimality operator for standard Q learning. When $K = 1$ and $f_1(\mathbf{a}) = \mu(\mathbf{a}|\mathbf{s})$, where $\mu(\mathbf{a}|\mathbf{s})$ is the estimated behavior policy, $\mathcal{T}_{\mathcal{F}}$ becomes the Expected-Max Q-Learning (EMaQ) with sample size $N \to \infty$ [7].

To study the property of DKQ, we first show that $\mathcal{T}_{\mathcal{F}}$ still has a unique fixed point denoted by $Q^*_{\mathcal{F}}$ in Theorem 1.

Theorem 1. *For all domain knowledge $\mathcal{F}$ defined in Definition 1, $\mathcal{T}_{\mathcal{F}}$ is a contraction operator on the Banach space of functions defined over $\mathcal{S}\times\mathcal{A}$ and the supremum norm $\|\cdot\|_\infty$. Thus, $\mathcal{T}_{\mathcal{F}}$ has a unique fixed point, denoted by $Q^*_{\mathcal{F}}$.*

Since the unique fixed point exists, we can still use existing approaches to learn a function approximation of $Q^*_{\mathcal{F}}$. However, we also need a policy derived from the learned Q function to make predictions at test time. For this purpose, we construct a policy $\pi^*_{\mathcal{F}}(\cdot|\mathbf{s})$ based on $\mathcal{F}$ and $Q^*_{\mathcal{F}}$, and show that its action value function is $Q^*_{\mathcal{F}}$ in Theorem 2.

Theorem 2. *Let $\pi^*_{\mathcal{F}}(\cdot|\mathbf{s})$ be the policy that selects $\arg\max_{\mathbf{a}\in supp(f_i)} Q^*_{\mathcal{F}}(\mathbf{s},\mathbf{a})$ with probability α_i and selects other actions with 0 probability. Then $Q^*_{\mathcal{F}}$ is the action-value function of $\pi^*_{\mathcal{F}}(\cdot|\mathbf{s})$.*

Note that if $\arg\max_{\mathbf{a}\in supp(f_i)} Q^*_{\mathcal{F}}(\mathbf{s},\mathbf{a})$ overlaps with $\arg\max_{\mathbf{a}\in supp(f_j)} Q^*_{\mathcal{F}}(\mathbf{s},\mathbf{a})$ where $i \neq j$, $\pi^*_{\mathcal{F}}(\cdot|\mathbf{s})$ selects these actions with probability $\alpha_i + \alpha_j$.

$\mathcal{T}_{\mathcal{F}}$ by definition is a conservative operator compared to $\mathcal{T}$. Lemma 1 formally describes this property. This property also leads to a conservative fixed point. Specifically, Q^* is lower bounded by $Q^*_{\mathcal{F}}$ everywhere on $\mathcal{S}\times\mathcal{A}$ (Theorem 3).

Lemma 1. *$\forall Q_1, Q_2 : \mathcal{S}\times\mathcal{A}\to\mathbb{R}$, if $\forall \mathbf{s}\in\mathcal{S}, \mathbf{a}\in\mathcal{A}$, $Q_1(\mathbf{s},\mathbf{a}) \le Q_2(\mathbf{s},\mathbf{a})$, then we have*

$$\forall \mathbf{s}\in\mathcal{S}, \mathbf{a}\in\mathcal{A}, \mathcal{T}_{\mathcal{F}}Q_1(\mathbf{s},\mathbf{a}) \le \mathcal{T}Q_2(\mathbf{s},\mathbf{a}).$$

Theorem 3. *$\forall \mathbf{s}\in\mathcal{S}, \mathbf{a}\in\mathcal{A}$, we have $Q^*_{\mathcal{F}}(\mathbf{s},\mathbf{a}) \le Q^*(\mathbf{s},\mathbf{a})$.*

Inspired by [7], we also analyze the sub-optimality bounds for DKQ, shown in Theorem 4. Theorem 4 indicates that the sub-optimality is bounded by the weighted sum of the gap of Q values on $\mathcal{A}$ and $supp(f_i)$, where the weights come from the knowledge $\mathcal{F}$. The conclusion applies for not only $Q^*_{\mathcal{F}}$ but also Q^* (details in Theorem 4). Of note, in the case of good domain knowledge, the α_i assigned to important action group is large, while those assigned to other groups are small or even close to zero. Thus, the bounds are small and $Q^*_{\mathcal{F}}$ is close to Q^*.

Theorem 4. *The sub-optimality of $Q^*_{\mathcal{F}}$ is bounded as follows:*

$$\|Q^*_{\mathcal{F}} - Q^*\|_\infty \le \frac{\gamma}{1-\gamma}\min\left(\sum_{i=1}^{K}\alpha_i \max_{\mathbf{s}}\Delta_{\mathcal{F},i}(\mathbf{s}), \sum_{i=1}^{K}\alpha_i \max_{\mathbf{s}}\Delta'_{\mathcal{F},i}(\mathbf{s})\right), \tag{3}$$

where $\Delta_{\mathcal{F},i}(\mathbf{s}) \coloneqq \max_{\mathbf{a}\in\mathcal{A}} Q^*_{\mathcal{F}}(\mathbf{s},\mathbf{a}) - \max_{\mathbf{a}\in supp(f_i)} Q^*_{\mathcal{F}}(\mathbf{s},\mathbf{a})$ and $\Delta'_{\mathcal{F},i}(\mathbf{s}) \coloneqq \max_{\mathbf{a}\in\mathcal{A}} Q^*(\mathbf{s},\mathbf{a}) - \max_{\mathbf{a}\in supp(f_i)} Q^*(\mathbf{s},\mathbf{a})$.

4.2 Fitted Iteration for DKQ

In this subsection, we describe a function approximation of DKQ through fitted iterations [19]. Fitted iteration is easy to implement and can utilize different function approximators. The Domain Knowledge Guided Fitted Q Iteration is listed in Algorithm 1.

As described in [21], fitted iteration is not guaranteed to converge. However, it converges to an l^∞ ball around Q^*, when the projection error at each iteration is uniformly bounded (Theorem B.1 in [21]). In Theorem 5, we show that this conclusion still holds for DKQ.

Theorem 5. *If the error of function approximation at each iteration is uniformly bounded by δ, that is $\forall n$, $\|\widehat{Q}^{(n+1)}_{\mathcal{F}} - \mathcal{T}_{\mathcal{F}}\widehat{Q}^{(n)}_{\mathcal{F}}\|_\infty \le \delta$, the error in the final solution of Algorithm 1 is bounded as follows:*

$$\lim_{n\to\infty}\|\widehat{Q}^{(n)}_{\mathcal{F}} - Q^*_{\mathcal{F}}\|_\infty \le \frac{\delta}{1-\gamma}. \tag{4}$$

Algorithm 1 $DK - FQI(\mathbf{f}, \boldsymbol{\alpha}, \gamma, \mathcal{D}, N)$

Initialize $\widehat{Q}_0 = \mathbf{0}$
for $t = 1, \ldots, T$ **do**
  $\mathbf{x}_t \leftarrow (\mathbf{s}_t, \mathbf{a}_t)$
end for
for $n = 1, \ldots, N$ **do**
  Build the training set $\mathcal{D}^{(n)} = \{(\mathbf{x}_t, y_t^{(n)})\}_{t=0}^{T}$ by setting each $y_t^{(n)}$ as follows:

$$y_t^{(n)} \leftarrow r_t + \gamma \sum_{i=1}^{K} \alpha_i \max_{\mathbf{a}' \in supp(f_i)} \widehat{Q}_{\mathcal{F}}^{(n-1)}(\mathbf{s}'_t, \mathbf{a}')$$

  $\widehat{Q}_{\mathcal{F}}^{(n)} \leftarrow regression\_training(\mathcal{D}^{(n)})$
end for
return $\widehat{Q}_{\mathcal{F}}^{(N)}$

As suggested by previous studies [5, 6], a key challenge in offline RL is unrealistic estimation of Q values for "out-of-distribution" actions. Although this is not the focus of this paper, we still demonstrate theoretically that good domain knowledge $\mathcal{F}$ can mitigate the problem of overestimation. Specifically, through $\mathcal{F}$, DKQ restricts the actions being selected and its impact. Depending on the application, it is possible for domain knowledge to contain information about the behavior policies that generated the historical data. In the presence of such domain knowledge, DKQ can avoid sampling unseen actions outside the behavior distribution. Meanwhile, DKQ does not introduce additional function approximation for the behavior policy, and thus further reduces errors.

5 Case Study: Suggest Next Best Node to Grow Family Tree with DKQ

In this section, we introduce a case study with DKQ. As described in the Introduction, we aim to help people grow their family trees by improving a genealogy service - a website or app where users can search for historical records, such as census records and birth certificates, to understand their heritage and family history. We formulate this problem as a RL model because family tree building is a sequential task. Users first create a node with basic information, and then use the genealogy service to find supporting documents for that node. Information discovered from relevant documents can inspire users to create the next node. Many people interested in family history lack genealogical research skills and reach a dead end when building family trees. Users may break through this barrier by exploring nodes that the genealogy service still has records for. However, it is challenging to locate those nodes, because a family tree can contain hundreds of nodes and users may not know which ones still have undiscovered documents. We can provide users with guidance for tree building using two methods: heuristic approaches based on domain knowledge and data-driven approaches. With DKQ, we can construct a solution that leverages both data and domain knowledge. Domain knowledge can come from consultation with professional genealogists, customer interviews and data analytics. In the rest of this section, we demonstrate in detail how DKQ utilizes available domain knowledge to suggest Next Best Node.

5.1 Data and features

For this case study, we used real data collected by a genealogy company with customer consent. We define trajectory as a sequence of nodes that a user interacted with. The data contains trajectories within a month from 1,000 family trees, with a total of 95,363 steps. We formulate the problem with DKQ by defining state, action, reward and domain knowledge as follows.

Reward The goal is to guide users to the node where new information is most likely to be discovered. If the user finds a record/document and believes that it belongs to a person (represented by a node on the family tree) in his/her family, then he/she typically saves the record to tree. We consider a record saved to tree as a "successful discovery" and assign a high reward score (10 points) as ground truth. If the user explores the node but saves no documents, we assign a lower positive reward (1 point).

State features The state is characterized by both user status and tree status, including user's product engagement level, number of nodes in tree, and content availability. The content availability feature estimates how many relevant documents exist in the genealogy service database for a particular tree.

Action features An action is a node in the family tree at a given timestamp. It is represented by an array with the following dimensions: me-person flag, generation, direct relative flag, recently being

interacted, kinship, engagement history, and node status. Me person is the focus person of the tree, which is specified by the user. The me-person flag is a binary feature to indicate whether the node is me-person or not. Generation is the number of generations counted from me person. The direct relative flag indicates whether a node is on the direct line of kinship with me-person, such as parents, grandparents, child, grandchild. The feature "recently being interacted" is a flag to track the last 10 nodes modified by the user. Kinship includes the relationships of the target node with both me-person and recently interacted nodes. Engagement history is the aggregate engagement level of the target node up to a given timestamp. Node status captures the level of information completeness.

5.2 DKQ and baselines

We implemented the Fitted Iteration version of DKQ with CatBoost Regressor [22] as the regression model. We chose CatBoost because it is a state-of-the-art ensemble algorithm and is capable of handling tabular data and categorical features.

Domain knowledge To maximize the chance of discovery, we take users' current interest into consideration. Previous analysis suggests that users are likely to continue working on nodes that they recently interacted with and those around the last interacted node. This knowledge partially reflects the behavior policy, which are confirmed by our experimental results shown in the next section. Accordingly, we define domain knowledge as follows:

- $f_1(\mathbf{a}) = 1$ if $\mathbf{a}$ is one of the ten recently interacted nodes.
- $f_2(\mathbf{a}) = 1$ if $\mathbf{a}$ is within two steps, in terms of kinship, from the ten recently interacted nodes.
- $f_3(\mathbf{a}) = 1$ if $\mathbf{a}$ is within three generations around me-person.
- $f_4(\mathbf{a}) = 1$ if $\mathbf{a}$ is a direct relative of me-person.

Otherwise, $f_i(\mathbf{a}) = 0$ for $i = 1, 2, 3, 4$. Due to page limit, the list of parameter values $\boldsymbol{\alpha} = [\alpha_1, \alpha_2, \alpha_3, \alpha_4]$ for experiments in Table 1 is included in the Appendix. Note that the available domain knowledge only covers f_1 and f_2. The corresponding models are listed in line 1, 2, 5 in Table 1. We include f_3 and f_4 (Table 1, line 3, 4, 7-11) as variants to show how different domain knowledge affects the performance.

Table 1: Mean reward on actions where the learned policy agrees/disagrees with the behavior policy

	Experiment name	R_+	R_-
1	Ten recently interacted nodes	13.92	7.21
2	Around recently interacted nodes	13.27	6.82
3	Three generations around me-person	16.74	8.19
4	Direct relatives	14.91	7.91
5	Ten recently interacted nodes OR Around recently interacted nodes	14.04	7.26
6	Ten recently interacted nodes OR Three generations around me-person	15.27	7.75
7	Ten recently interacted nodes OR Direct relatives	14.97	7.59
8	Around recently interacted nodes OR Three generations around me-person	14.84	7.72
9	Around recently interacted nodes OR Direct relatives	14.83	7.63
10	Three generations around me-person OR Direct relatives	15.50	7.93
11	Ten recently interacted nodes OR Around recently interacted nodes OR Direct relatives	14.79	7.68
12	All nodes	15.91	7.85
13	Last interacted node (baseline)	8.41	8.13

Baselines We consider two baselines: Last Interacted Node (Table 1, line 13) and Q Learning Without Domain Knowledge (Table 1, line 12). The first one is a rule-based approach. Intuitively, it is safer, because it rarely recommends a node far from customer intent. However, it follows the user behavior and is not driven by the defined success metric. On the other hand, the second baseline is completely data-driven with no human guidance. It may result in worse online performance due to too many unseen actions and unrealistic value estimates in offline training.

5.3 Evaluation and results

The evaluation of offline RL is a well-known challenge, because the learned policy has not been rolled out online and the test data is still from the behavioral policy. The expected average reward

of the learned policy $\widehat{\pi}$ defined as $V(\widehat{\pi}) = \frac{1}{T}\sum_{t=1}^{T}\mathbb{E}_{\mathbf{s}\sim\mathcal{D}_t}[r(\mathbf{s}_t, \widehat{\pi}(\mathbf{s}_t))]$ is impossible to estimate, where $\mathcal{D}_t$ is the marginal distribution of data at step t. Moreover, for this specific problem, it is not feasible to collect data using some randomized policy. Inspired by [23], we simplify this estimator as the mean reward on actions where the model agrees with the data, represented by $R_+ = \frac{1}{T}\sum_{t=1}^{T}\mathbb{I}_{\mathbf{a}_t=\widehat{\pi}(\mathbf{s}_t)}r(\mathbf{s}_t, \mathbf{a}_t)$. Similarly, we define $R_- = \frac{1}{T}\sum_{t=1}^{T}\mathbb{I}_{\mathbf{a}_t\neq\widehat{\pi}(\mathbf{s}_t)}r(\mathbf{s}_t, \mathbf{a}_t)$. Intuitively, the idea is to associate the difference between the behavior policy and the learned policy with a desired outcome, which is the reward or discoveries in our case. The learned policy is expected to recommend high-reward actions in the behavioral data and deviate from low- or no-reward actions. If the learned policy $\widehat{\pi}$ is good, the reward should be high on actions where the behavior policy agrees with $\widehat{\pi}$, and low on actions where the behavior policy disagrees with $\widehat{\pi}$. Thus we examine if each learned policy has a high R_+ and a low R_- as shown in Table 1.

Figure 1 shows: 1. Considering "ratio of rewards", all data driven algorithms are evidently better than the rule-based baseline. 2. The "percentage of actions agreed" of rule-based baseline is substantially larger than others. This also verifies that the domain knowledge partially reflects the behavior policy. 3. The data driven models have similar performance with respect to "ratio of rewards", but differ in the "percentage of actions agreed". The baseline without domain knowledge represented by "All nodes" show only about 5% agreement with the data, which is caused by a large number of out-of-distribution actions. The three variants guided by domain knowledge, "Ten recently interacted nodes", "Around recently interacted nodes" and "Ten recently interacted nodes OR Around recently interacted nodes", hold the highest "percentage of actions agreed" and relatively high "ratio of rewards". Therefore, policies generated from these three variants better balance issue of unseen data and predicted reward.

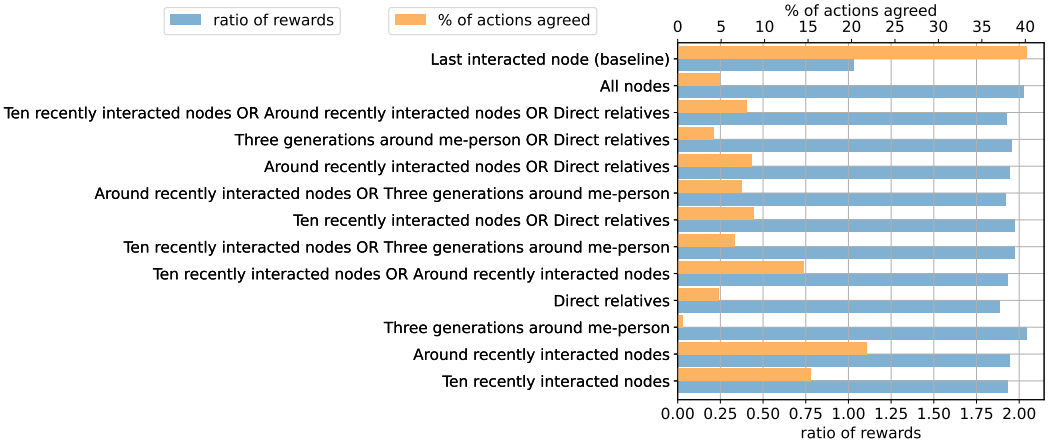

Figure 1: The ratio R_+/R_- and the proportion of the observed data agreed the learned policy.

6 Experiments on Benchmark Dataset

In this section, we further demonstrate the benefits of DKQ with a classic control problem, Acrobot [24, 25]. The acrobot system includes a two-link pendulum that hangs downwards initially. The goal is to swing the end of the lower link above a given height. Acrobot has a continuous state space with six dimensions, including $sin()$ and $cos()$ of the two joint angles and the joint angular velocities. There are three possible actions, which are applying -1, 0, or +1 torque on the joint between the two pendulum links. Reward is 0 for reaching the goal and -1 otherwise. We conducted experiments using the Intel Coach library (Apache License 2.0) [26] on an AWS r5a.2xlarge CPU instance.

6.1 DKQ with random action sampling and BCQ

For DKQ, each Q-update maximizes over a subset of actions that are pre-determined by domain knowledge. Since we do not have the domain knowledge for the Acrobot problem, we define the support of f_i in Equation (2) as a set of two random actions drawn from the three possible actions (-1, 0, 1). Both K and α are set to 1. This formulation becomes a Markov decision process with stochastic action sets as described in [27]. Work in [27] suggests that Q-learning with sampled action sets will converge to the optimal Q-function. We compare DKQ with random action sampling to BCQ

mentioned in Section 2. BCQ restricts the action space by training a generative model to produce only actions that were seen before [5]. Action sampling in BCQ forces the offline agent to stay close to the previous behavioral policy. Both DKQ and BCQ in this experiment employed Double DQN [28] for function approximation. The same neural network parameters were used in both methods. In comparison to DKQ, there are two potential issues with applying approaches like BCQ in practice. Training a model to emulate the behavioral policy may introduce additional approximation errors and will increase computation time [7, 29]. Although certain choices of the generative model for estimating behavioral policies, such as autoregressive models, have been shown to perform better, they may be even slower and require extra engineering effort to speed up [7]. We examined those two issues by evaluating the computation time and the performance of BCQ and DKQ with random action sampling. We assess the performance using two off-policy evaluation estimators, namely weighted importance sampling [30] and sequential doubly robust estimator [31].

6.2 Results

We used a small and simple dataset containing 30 episodes with an average of 325 transitions per episode. We repeated both BCQ and DKQ 30 times with 100 epochs in each run. The output bias was initialized to -100. We evaluated the wall time for every 100 epochs and the best-performing agent of each run. As shown in Figure 2, the performances of BCQ and DKQ with random action sampling are comparable for both weighted importance sampling and sequential doubly robust estimators. However, DKQ with random action sampling is 4.5 times faster. The median computation time is 353 seconds and 77 seconds for BCQ and DKQ, respectively.

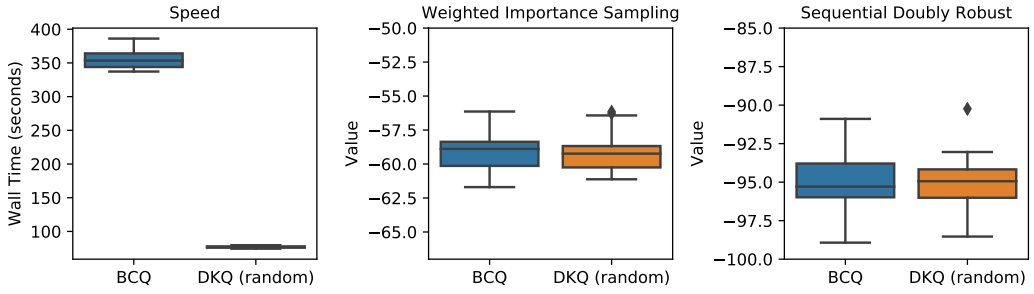

Figure 2: Comparison between BCQ and DKQ with random action sampling using weighted importance sampling and sequential doubly robust off-policy estimators.

Although the computation time depends on implementation and device, BCQ has been shown previously to be much slower than DQN [29]. Given the substantial speed difference between BCQ and DKQ with random action sampling on such a small and simple benchmark set, we expect DKQ to be a more efficient offline RL framework than BCQ and similar methods in real-world applications. Real-world data is more complex and scaling up is typically needed, but optimization can be expensive. Further, companies usually have domain knowledge about their product and services, which may help break the trade-off between training time and performance.

7 Conclusions

This paper aims to further bridge the gap between offline RL and broad real-world applications in the presence of domain knowledge. We propose a framework, Domain Knowledge guided Q learning (DKQ), that flexibly incorporates domain knowledge into offline policy learning. We not only theoretically study the properties of DKQ in the tabular setting, but also provide a real-world case study, which applies DKQ to help users with family tree building for the first time. The results show that DKQ lowers the risk of overestimation on unseen data while preserving high offline performance. A potential issue is that DKQ relies on domain knowledge, regardless of the knowledge quality. Nevertheless, our findings suggest that DKQ is sound from the RL perspective and it is an effective and efficient method for real-world applications where domain knowledge is readily available.

References

[1] Floris den Hengst, Eoin Martino Grua, Ali el Hassouni, and Mark Hoogendoorn. Reinforcement learning for personalization: A systematic literature review. *Data Science*, 3(2):107–147, 2020.

[2] Jun Hou, Hua Li, Jinwen Hu, Chunhui Zhao, Yaning Guo, Sijia Li, and Quan Pan. A review of the applications and hotspots of reinforcement learning. In *2017 IEEE International Conference on Unmanned Systems (ICUS)*, pages 506–511, 2017.

[3] Aviral Kumar, Justin Fu, Matthew Soh, George Tucker, and Sergey Levine. Stabilizing off-policy q-learning via bootstrapping error reduction. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

[4] Sergey Levine, Aviral Kumar, George Tucker, and Justin Fu. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *arXiv preprint arXiv:2005.01643*, 2020.

[5] Scott Fujimoto, David Meger, and Doina Precup. Off-policy deep reinforcement learning without exploration. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 2052–2062, 2019.

[6] Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative q-learning for offline reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 33, pages 1179–1191, 2020.

[7] Seyed Kamyar Seyed Ghasemipour, Dale Schuurmans, and Shixiang Shane Gu. Emaq: Expected-max q-learning operator for simple yet effective offline and online RL. *arXiv preprint arXiv:2007.11091*, 2020.

[8] Aaron Sonabend, Junwei Lu, Leo Anthony Celi, Tianxi Cai, and Peter Szolovits. Expert-supervised reinforcement learning for offline policy learning and evaluation. In *Advances in Neural Information Processing Systems*, volume 33, pages 18967–18977, 2020.

[9] Michael Fire, Thomas Chesney, and Yuval Elovici. Quantitative analysis of genealogy using digitised family trees. *arXiv preprint arXiv:1408.5571*, 2014.

[10] Kinga Marton, Katalin Nagy, and Alin Suciu. Collaborative genealogy tree in the cloud. In *2013 11th RoEduNet International Conference*, pages 1–5, 2013.

[11] James P Philips and Nasseh Tabrizi. Historical document processing: Historical document processing: A survey of techniques, tools, and trends. *arXiv preprint arXiv:2002.06300*, 2020.

[12] Paul Covington, Jay Adams, and Emre Sargin. Deep neural networks for youtube recommendations. In *Proceedings of the 10th ACM Conference on Recommender Systems*, RecSys '16, page 191–198, 2016.

[13] Lihong Li, Remi Munos, and Csaba Szepesvari. Toward minimax off-policy value estimation. In *Proceedings of the 18th International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2015.

[14] Yifan Wu, George Tucker, and Ofir Nachum. Behavior regularized offline reinforcement learning. *arXiv preprint arXiv:1911.11361*, 2019.

[15] Scott Fujimoto, Herke van Hoof, and David Meger. Addressing function approximation error in actor-critic methods. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 1587–1596, 2018.

[16] Rishabh Agarwal, Dale Schuurmans, and Mohammad Norouzi. An optimistic perspective on offline reinforcement learning. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 104–114, 2020.

[17] Peter L. Bartlett and Ambuj Tewari. Regal: A regularization based algorithm for reinforcement learning in weakly communicating mdps. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*, UAI '09, page 35–42, 2009.

[18] Andrew Silva and Matthew Gombolay. Neural-encoding human experts' domain knowledge to warm start reinforcement learning. *arXiv preprint arXiv:1902.06007*, 2019.

[19] Damien Ernst, Pierre Geurts, and Louis Wehenkel. Tree-based batch mode reinforcement learning. *Journal of Machine Learning Research*, 6:503–556, 2005.

[20] Gerhard Neumann and Jan Peters. Fitted q-iteration by advantage weighted regression. In *Advances in Neural Information Processing Systems*, volume 21, 2009.

[21] Justin Fu, Aviral Kumar, Matthew Soh, and Sergey Levine. Diagnosing bottlenecks in deep q-learning algorithms. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 2021–2030, 2019.

[22] Liudmila Prokhorenkova, Gleb Gusev, Aleksandr Vorobev, Anna Veronika Dorogush, and Andrey Gulin. Catboost: unbiased boosting with categorical features. In *Advances in Neural Information Processing Systems*, volume 31, 2018.

[23] Omer Gottesman, Fredrik Johansson, Joshua Meier, Jack Dent, Donghun Lee, Srivatsan Srinivasan, Linying Zhang, Yi Ding, David Wihl, Xuefeng Peng, et al. Evaluating reinforcement learning algorithms in observational health settings. *arXiv preprint arXiv:1805.12298*, 2018.

[24] Ali Mousavi, Lihong Li, Qiang Liu, and Denny Zhou. Black-box off-policy estimation for infinite-horizon reinforcement learning. In *Proceedings of the Eighth International Conference on Learning Representations*, 2020.

[25] Lex Weaver and Nigel Tao. The optimal reward baseline for gradient-based reinforcement learning. In *Proceedings of the Seventeenth Conference on Uncertainty in Artificial Intelligence*, UAI'01, page 538–545, 2001.

[26] Itai Caspi, Gal Leibovich, Gal Novik, and Shadi Endrawis. Reinforcement learning coach, 2017. URL `https://doi.org/10.5281/zenodo.1134899`.

[27] Craig Boutilier, Alon Cohen, Avinatan Hassidim, Yishay Mansour, Ofer Meshi, Martin Mladenov, and Dale Schuurmans. Planning and learning with stochastic action sets. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*, pages 4674–4682, 2018.

[28] Hado van Hasselt, Arthur Guez, and David Silver. Deep reinforcement learning with double q-learning. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, AAAI'16, page 2094–2100, 2016.

[29] Aayam Kumar Shrestha, Stefan Lee, Prasad Tadepalli, and Alan Fern. Deepaveragers: Offline reinforcement learning by solving derived non-parametric {mdp}s. In *International Conference on Learning Representations*, 2021.

[30] Philip Thomas and Emma Brunskill. Data-efficient off-policy policy evaluation for reinforcement learning. In *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 2139–2148, 2016.

[31] Nan Jiang and Lihong Li. Doubly robust off-policy value evaluation for reinforcement learning. In *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 652–661, 2016.

A Proofs

A.1 Proof of Theorem 1

Proof. $\forall Q_1, Q_2 : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$,

$$\|\mathcal{T}_{\mathcal{F}} Q_1 - \mathcal{T}_{\mathcal{F}} Q_2\|_\infty \tag{5}$$

$$= \|r(\mathbf{s},\mathbf{a}) + \gamma \mathbb{E}_{\mathbf{s}' \sim P(\mathbf{s}'|\mathbf{s},\mathbf{a})} \big[\sum_{i=1}^{K} \alpha_i \max_{\mathbf{a}' \in supp(f_i)} Q_1(\mathbf{s}', \mathbf{a}') \big] \tag{6}$$

$$- r(\mathbf{s},\mathbf{a}) - \gamma \mathbb{E}_{\mathbf{s}' \sim P(\mathbf{s}'|\mathbf{s},\mathbf{a})} \big[\sum_{i=1}^{K} \alpha_i \max_{\mathbf{a}' \in supp(f_i)} Q_2(\mathbf{s}', \mathbf{a}') \big] \|_\infty \tag{7}$$

$$= \gamma \max_{\mathbf{s},\mathbf{a}} \left| \mathbb{E}_{\mathbf{s}' \sim P(\mathbf{s}'|\mathbf{s},\mathbf{a})} \left[\sum_{i=1}^{K} \alpha_i \max_{\mathbf{a}' \in supp(f_i)} Q_1(\mathbf{s}', \mathbf{a}') - \sum_{i=1}^{K} \alpha_i \max_{\mathbf{a}' \in supp(f_i)} Q_2(\mathbf{s}', \mathbf{a}') \right] \right| \tag{8}$$

$$\leq \gamma \max_{\mathbf{s},\mathbf{a}} \mathbb{E}_{\mathbf{s}' \sim P(\mathbf{s}'|\mathbf{s},\mathbf{a})} \left[\left| \sum_{i=1}^{K} \alpha_i \max_{\mathbf{a}' \in supp(f_i)} Q_1(\mathbf{s}', \mathbf{a}') - \sum_{i=1}^{K} \alpha_i \max_{\mathbf{a}' \in supp(f_i)} Q_2(\mathbf{s}', \mathbf{a}') \right| \right] \tag{9}$$

$$= \gamma \max_{\mathbf{s},\mathbf{a}} \mathbb{E}_{\mathbf{s}' \sim P(\mathbf{s}'|\mathbf{s},\mathbf{a})} \left[\left| \sum_{i=1}^{K} \alpha_i \big(\max_{\mathbf{a}' \in supp(f_i)} Q_1(\mathbf{s}', \mathbf{a}') - \max_{\mathbf{a}' \in supp(f_i)} Q_2(\mathbf{s}', \mathbf{a}') \big) \right| \right] \tag{10}$$

$$\leq \gamma \max_{\mathbf{s},\mathbf{a}} \mathbb{E}_{\mathbf{s}' \sim P(\mathbf{s}'|\mathbf{s},\mathbf{a})} \left[\sum_{i=1}^{K} \alpha_i \left| \max_{\mathbf{a}' \in supp(f_i)} Q_1(\mathbf{s}', \mathbf{a}') - \max_{\mathbf{a}' \in supp(f_i)} Q_2(\mathbf{s}', \mathbf{a}') \right| \right] \tag{11}$$

$$\leq \gamma \max_{\mathbf{s},\mathbf{a}} \mathbb{E}_{\mathbf{s}' \sim P(\mathbf{s}'|\mathbf{s},\mathbf{a})} \left[\sum_{i=1}^{K} \alpha_i \max_{\mathbf{a}' \in supp(f_i)} \left| Q_1(\mathbf{s}', \mathbf{a}') - Q_2(\mathbf{s}', \mathbf{a}') \right| \right] \tag{12}$$

$$\leq \gamma \max_{\mathbf{s},\mathbf{a}} \mathbb{E}_{\mathbf{s}' \sim P(\mathbf{s}'|\mathbf{s},\mathbf{a})} \left[\sum_{i=1}^{K} \alpha_i \max_{\mathbf{a}' \in supp(f_i)} \|Q_1 - Q_2\|_\infty \right] \tag{13}$$

$$= \gamma \sum_{i=1}^{K} \alpha_i \|Q_1 - Q_2\|_\infty \tag{14}$$

$$= \gamma \|Q_1 - Q_2\|_\infty \tag{15}$$

Step (12) is because $\forall g_1, g_2 : \mathcal{A} \to \mathbb{R}$, we have

$$|\max_{\mathbf{a}} g_1(\mathbf{a}) - \max_{\mathbf{a}} g_2(\mathbf{a})| \leq \max_{\mathbf{a}} |g_1(\mathbf{a}) - g_2(\mathbf{a})|. \tag{16}$$

Specifically, denote $\mathbf{a}_1 = \arg\max_{\mathbf{a}} g_1(\mathbf{a})$, $\mathbf{a}_2 = \arg\max_{\mathbf{a}} g_2(\mathbf{a})$, then we have

$$\max_{\mathbf{a}} |g_1(\mathbf{a}) - g_2(\mathbf{a})| \geq g_1(\mathbf{a}_1) - g_2(\mathbf{a}_1) \geq g_1(\mathbf{a}_1) - g_2(\mathbf{a}_2) = \max_{\mathbf{a}} g_1(\mathbf{a}) - \max_{\mathbf{a}} g_2(\mathbf{a}) \tag{17}$$

$$\max_{\mathbf{a}} |g_2(\mathbf{a}) - g_1(\mathbf{a})| \geq g_2(\mathbf{a}_2) - g_1(\mathbf{a}_2) \geq g_2(\mathbf{a}_2) - g_1(\mathbf{a}_1) = \max_{\mathbf{a}} g_2(\mathbf{a}) - \max_{\mathbf{a}} g_1(\mathbf{a}) \tag{18}$$

Combine (17) and (18), we have (16).

Step (15) is because $\sum_{i=1}^{K} \alpha_i = 1$. Thus, $\mathcal{T}_{\mathcal{F}}$ is a contraction operator, and there exists a unique fixed point denoted as $Q^*_{\mathcal{F}}$. And we have $Q^*_{\mathcal{F}} = \mathcal{T}_{\mathcal{F}} Q^*_{\mathcal{F}}$. □

A.2 Proof of Theorem 2

Proof.

$$Q^*_{\mathcal{F}}(\mathbf{s},\mathbf{a}) = \mathcal{T}_{\mathcal{F}} Q^*_{\mathcal{F}}(\mathbf{s},\mathbf{a}) \tag{19}$$

$$= r(\mathbf{s},\mathbf{a}) + \gamma \mathbb{E}_{\mathbf{s}' \sim P(\mathbf{s}'|\mathbf{s},\mathbf{a})} \big[\sum_{i=1}^{K} \alpha_i \max_{\mathbf{a}' \in supp(f_i)} Q^*_{\mathcal{F}}(\mathbf{s}', \mathbf{a}') \big] \tag{20}$$

$$= r(\mathbf{s},\mathbf{a}) + \gamma \mathbb{E}_{\mathbf{s}' \sim P(\mathbf{s}'|\mathbf{s},\mathbf{a})} \mathbb{E}_{\mathbf{a}' \sim \pi^*_{\mathcal{F}}(\mathbf{a}'|\mathbf{s}')} \big[Q^*_{\mathcal{F}}(\mathbf{s}', \mathbf{a}') \big] \tag{21}$$

Step (19) is because $Q^*_{\mathcal{F}}$ is the fixed point of $\mathcal{T}_{\mathcal{F}}$. In step (21), we get $\mathbb{E}_{\mathbf{a}'\sim\pi^*_{\mathcal{F}}(\mathbf{a}'|\mathbf{s}')}\big[Q^*_{\mathcal{F}}(\mathbf{s}',\mathbf{a}')\big] = \sum_{i=1}^{K}\alpha_i \max_{\mathbf{a}'\in supp(f_i)} Q^*_{\mathcal{F}}(\mathbf{s}',\mathbf{a}')$ from the definition of $\pi^*_{\mathcal{F}}(\cdot|\mathbf{s})$.

Thus $Q^*_{\mathcal{F}}$ satisfies the Bellman expectation equation of policy $\pi^*_{\mathcal{F}}$. Because the Bellman expectation operator has unique fixed point which is the action-value function of the policy, then $Q^*_{\mathcal{F}}$ is the action-value function of $\pi^*_{\mathcal{F}}$. □

A.3 Proof of Lemma 1

Proof. $\forall Q_1, Q_2 : \mathcal{S}\times\mathcal{A}\to\mathbb{R}$ such that $\forall \mathbf{s}\in\mathcal{S}, \mathbf{a}\in\mathcal{A}$, $Q_1(\mathbf{s},\mathbf{a}) \le Q_2(\mathbf{s},\mathbf{a})$, then we have $\forall \mathbf{s}\in\mathcal{S}, \mathbf{a}\in\mathcal{A}$,

$$\mathcal{T}Q_2(\mathbf{s},\mathbf{a}) - \mathcal{T}_{\mathcal{F}}Q_1(\mathbf{s},\mathbf{a}) \tag{22}$$

$$= \gamma\mathbb{E}_{\mathbf{s}'\sim P(\mathbf{s}'|\mathbf{s},\mathbf{a})}\big[\max_{\mathbf{a}'\in\mathcal{A}} Q_2(\mathbf{s}',\mathbf{a}')\big] - \gamma\mathbb{E}_{\mathbf{s}'\sim P(\mathbf{s}'|\mathbf{s},\mathbf{a})}\big[\sum_{i=1}^{K}\alpha_i \max_{\mathbf{a}'\in supp(f_i)} Q_1(\mathbf{s}',\mathbf{a}')\big] \tag{23}$$

$$= \gamma\mathbb{E}_{\mathbf{s}'\sim P(\mathbf{s}'|\mathbf{s},\mathbf{a})}\big[\max_{\mathbf{a}'\in\mathcal{A}} Q_2(\mathbf{s}',\mathbf{a}') - \sum_{i=1}^{K}\alpha_i \max_{\mathbf{a}'\in supp(f_i)} Q_1(\mathbf{s}',\mathbf{a}')\big] \tag{24}$$

$$= \gamma\mathbb{E}_{\mathbf{s}'\sim P(\mathbf{s}'|\mathbf{s},\mathbf{a})}\big[\sum_{i=1}^{K}\alpha_i \max_{\mathbf{a}'\in\mathcal{A}} Q_2(\mathbf{s}',\mathbf{a}') - \sum_{i=1}^{K}\alpha_i \max_{\mathbf{a}'\in supp(f_i)} Q_1(\mathbf{s}',\mathbf{a}')\big] \tag{25}$$

$$= \gamma\mathbb{E}_{\mathbf{s}'\sim P(\mathbf{s}'|\mathbf{s},\mathbf{a})}\big[\sum_{i=1}^{K}\alpha_i\big(\max_{\mathbf{a}'\in\mathcal{A}} Q_2(\mathbf{s}',\mathbf{a}') - \max_{\mathbf{a}'\in supp(f_i)} Q_1(\mathbf{s}',\mathbf{a}')\big)\big] \tag{26}$$

$$\ge \gamma\mathbb{E}_{\mathbf{s}'\sim P(\mathbf{s}'|\mathbf{s},\mathbf{a})}\big[\sum_{i=1}^{K}\alpha_i\big(\max_{\mathbf{a}'\in\mathcal{A}} Q_1(\mathbf{s}',\mathbf{a}') - \max_{\mathbf{a}'\in supp(f_i)} Q_1(\mathbf{s}',\mathbf{a}')\big)\big] \tag{27}$$

$$\ge 0 \tag{28}$$

Step (25) is because $\sum_{i=1}^{K}\alpha_i = 1$ and $\alpha_i \ge 0$.

Step (27) is because the $Q_1(\mathbf{s},\mathbf{a}) \le Q_2(\mathbf{s},\mathbf{a})$ ($\forall \mathbf{s},\mathbf{a}$).

In step (28), $supp(f_i)\subseteq\mathcal{A}$, thus $\forall \mathbf{s}'\in\mathcal{S}, \max_{\mathbf{a}'\in\mathcal{A}} Q_1(\mathbf{s}',\mathbf{a}') \ge \max_{\mathbf{a}'\in supp(f_i)} Q_1(\mathbf{s}',\mathbf{a}')$.

Therefore, $\forall \mathbf{s},\mathbf{a}, \mathcal{T}_{\mathcal{F}}Q_1(\mathbf{s},\mathbf{a}) \le \mathcal{T}Q_2(\mathbf{s},\mathbf{a})$. □

A.4 Proof of Theorem 3

Proof. Let $(\mathcal{T})^n$ and $(\mathcal{T}_{\mathcal{F}})^n$ respectively denote applying $\mathcal{T}$ and $\mathcal{T}_{\mathcal{F}}$ by n times.

If for some $n\in\mathbb{N}$, $\forall \mathbf{s}\in\mathcal{S}, \mathbf{a}\in\mathcal{A}$, $(\mathcal{T}_{\mathcal{F}})^n Q(\mathbf{s},\mathbf{a}) \le (\mathcal{T})^n Q(\mathbf{s},\mathbf{a})$, then we have

$$(\mathcal{T}_{\mathcal{F}})^{n+1}Q(\mathbf{s},\mathbf{a}) = \mathcal{T}_{\mathcal{F}}((\mathcal{T}_{\mathcal{F}})^n Q(\mathbf{s},\mathbf{a})) \tag{29}$$

$$\le \mathcal{T}((\mathcal{T})^n Q(\mathbf{s},\mathbf{a})) \tag{30}$$

$$= (\mathcal{T})^{n+1}Q(\mathbf{s},\mathbf{a}), \tag{31}$$

where (30) is from Lemma 1.

And because $(\mathcal{T}_{\mathcal{F}})^0 Q(\mathbf{s},\mathbf{a}) \le (\mathcal{T})^0 Q(\mathbf{s},\mathbf{a})$, through induction, we have that $\forall \mathbf{s}\in\mathcal{S}, \mathbf{a}\in\mathcal{A}, \forall n\in\mathbb{N}$,

$$(\mathcal{T}_{\mathcal{F}})^n Q(\mathbf{s},\mathbf{a}) \le (\mathcal{T})^n Q(\mathbf{s},\mathbf{a}). \tag{32}$$

We know that $\lim_{n\to\infty}(\mathcal{T}_{\mathcal{F}})^n Q = Q^*_{\mathcal{F}}$ and $\lim_{n\to\infty}(\mathcal{T})^n Q = Q^*$. Thus, let $n\to\infty$ on both sides of inequality (32), we get $Q^*_{\mathcal{F}}(\mathbf{s},\mathbf{a}) \le Q^*(\mathbf{s},\mathbf{a})$ ($\forall \mathbf{s},\mathbf{a}$). □

A.5 Proof of Theorem 4

Proof. The proof is very similar to the proof of Theorem 3.5 of EMaQ [7]. According to the proof of Theorem 1, $\forall Q_1, Q_2$, we have $\|\mathcal{T}_{\mathcal{F}}Q_1 - \mathcal{T}_{\mathcal{F}}Q_2\|_\infty \leq \gamma\|Q_1 - Q_2\|_\infty$. Thus,

$$\begin{aligned}\|(\mathcal{T}_{\mathcal{F}})^{n+1}Q - (\mathcal{T}_{\mathcal{F}})^nQ\|_\infty &= \|\mathcal{T}_{\mathcal{F}}\big((\mathcal{T}_{\mathcal{F}})^nQ\big) - \mathcal{T}_{\mathcal{F}}\big((\mathcal{T}_{\mathcal{F}})^{n-1}Q\big)\|_\infty \\ &\leq \gamma\|(\mathcal{T}_{\mathcal{F}})^nQ - (\mathcal{T}_{\mathcal{F}})^{n-1}Q\|_\infty \\ &\leq \gamma^2\|(\mathcal{T}_{\mathcal{F}})^{n-1}Q - (\mathcal{T}_{\mathcal{F}})^{n-2}Q\|_\infty \\ &\cdots\cdots \\ &\leq \gamma^n\|(\mathcal{T}_{\mathcal{F}})Q - Q\|_\infty.\end{aligned}$$

That is, $\forall Q : \mathcal{S} \times \mathcal{A} \to \mathbb{R}, n \in \mathbb{N}$, we have

$$\|(\mathcal{T}_{\mathcal{F}})^{n+1}Q - (\mathcal{T}_{\mathcal{F}})^nQ\|_\infty \leq \gamma^n\|\mathcal{T}_{\mathcal{F}}Q - Q\|_\infty. \tag{33}$$

Similarly, $\forall Q : \mathcal{S} \times \mathcal{A} \to \mathbb{R}, n \in \mathbb{N}$, we have

$$\|(\mathcal{T})^{n+1}Q - (\mathcal{T})^nQ\|_\infty \leq \gamma^n\|\mathcal{T}Q - Q\|_\infty. \tag{34}$$

Then we derive both parts of the bound separately. We will prove that $\|Q^*_{\mathcal{F}} - Q^*\|_\infty \leq \frac{\gamma}{1-\gamma}\sum_{i=1}^K \alpha_i \max_{\mathbf{s}} \Delta_{\mathcal{F},i}(\mathbf{s})$ in Part 1, and prove $\|Q^*_{\mathcal{F}} - Q^*\|_\infty \leq \frac{\gamma}{1-\gamma}\sum_{i=1}^K \alpha_i \max_{\mathbf{s}} \Delta'_{\mathcal{F},i}(\mathbf{s})$ in Part 2.

Part 1: Proof of $\|Q^*_{\mathcal{F}} - Q^*\|_\infty \leq \frac{\gamma}{1-\gamma}\sum_{i=1}^K \alpha_i \max_{\mathbf{s}} \Delta_{\mathcal{F},i}(\mathbf{s})$.

$$\begin{aligned}\|Q^*_{\mathcal{F}} - Q^*\|_\infty &= \|Q^*_{\mathcal{F}} - \lim_{n\to\infty}(\mathcal{T})^nQ^*_{\mathcal{F}}\|_\infty && (35)\\ &= \|\lim_{n\to\infty}\big(Q^*_{\mathcal{F}} - (\mathcal{T})^nQ^*_{\mathcal{F}}\big)\|_\infty && (36)\\ &= \|\lim_{n\to\infty}\sum_{k=0}^{n-1}\big((\mathcal{T})^kQ^*_{\mathcal{F}} - (\mathcal{T})^{k+1}Q^*_{\mathcal{F}}\big)\|_\infty && (37)\\ &= \|\sum_{k=0}^{\infty}\big((\mathcal{T})^kQ^*_{\mathcal{F}} - (\mathcal{T})^{k+1}Q^*_{\mathcal{F}}\big)\|_\infty && (38)\\ &\leq \sum_{k=0}^{\infty}\|(\mathcal{T})^kQ^*_{\mathcal{F}} - (\mathcal{T})^{k+1}Q^*_{\mathcal{F}}\|_\infty && (39)\\ &\leq \sum_{k=0}^{\infty}\gamma^k\|Q^*_{\mathcal{F}} - \mathcal{T}Q^*_{\mathcal{F}}\|_\infty && (40)\\ &= \frac{1}{1-\gamma}\|Q^*_{\mathcal{F}} - \mathcal{T}Q^*_{\mathcal{F}}\|_\infty && (41)\\ & && (42)\end{aligned}$$

Step (35) is because $\lim_{n\to\infty}(T)^nQ^*_{\mathcal{F}} = Q^*$. Step (40) is from (34).

Then because $Q^*_{\mathcal{F}}$ is the fixed point of $\mathcal{T}_{\mathcal{F}}$,

$$\|Q^*_{\mathcal{F}} - Q^*\|_\infty \leq \frac{1}{1-\gamma}\|Q^*_{\mathcal{F}} - \mathcal{T}Q^*_{\mathcal{F}}\|_\infty \tag{43}$$

$$= \frac{1}{1-\gamma}\|\mathcal{T}_{\mathcal{F}}Q^*_{\mathcal{F}} - \mathcal{T}Q^*_{\mathcal{F}}\|_\infty \tag{44}$$

$$= \frac{1}{1-\gamma}\max_{\mathbf{s},\mathbf{a}}|\mathcal{T}_{\mathcal{F}}Q^*_{\mathcal{F}}(\mathbf{s},\mathbf{a}) - \mathcal{T}Q^*_{\mathcal{F}}(\mathbf{s},\mathbf{a})| \tag{45}$$

$$= \frac{1}{1-\gamma}\max_{\mathbf{s},\mathbf{a}}\big(\mathcal{T}Q^*_{\mathcal{F}}(\mathbf{s},\mathbf{a}) - \mathcal{T}_{\mathcal{F}}Q^*_{\mathcal{F}}(\mathbf{s},\mathbf{a})\big) \tag{46}$$

$$= \frac{1}{1-\gamma}\max_{\mathbf{s},\mathbf{a}}\big(\gamma\mathbb{E}_{\mathbf{s}'}\big[\max_{\mathbf{a}'\in\mathcal{A}}Q^*_{\mathcal{F}}(\mathbf{s}',\mathbf{a}')\big] - \gamma\mathbb{E}_{\mathbf{s}'}\big[\sum_{i=1}^{K}\alpha_i\max_{\mathbf{a}'\in supp(f_i)}Q^*_{\mathcal{F}}(\mathbf{s}',\mathbf{a}')\big]\big) \tag{47}$$

$$= \frac{\gamma}{1-\gamma}\max_{\mathbf{s},\mathbf{a}}\mathbb{E}_{\mathbf{s}'}\big[\max_{\mathbf{a}'\in\mathcal{A}}Q^*_{\mathcal{F}}(\mathbf{s}',\mathbf{a}') - \sum_{i=1}^{K}\alpha_i\max_{\mathbf{a}'\in supp(f_i)}Q^*_{\mathcal{F}}(\mathbf{s}',\mathbf{a}')\big] \tag{48}$$

$$= \frac{\gamma}{1-\gamma}\max_{\mathbf{s},\mathbf{a}}\mathbb{E}_{\mathbf{s}'}\big[\sum_{i=1}^{K}\alpha_i\max_{\mathbf{a}'\in\mathcal{A}}Q^*_{\mathcal{F}}(\mathbf{s}',\mathbf{a}') - \sum_{i=1}^{K}\alpha_i\max_{\mathbf{a}'\in supp(f_i)}Q^*_{\mathcal{F}}(\mathbf{s}',\mathbf{a}')\big] \tag{49}$$

$$= \frac{\gamma}{1-\gamma}\max_{\mathbf{s},\mathbf{a}}\mathbb{E}_{\mathbf{s}'}\big[\sum_{i=1}^{K}\alpha_i\big(\max_{\mathbf{a}'\in\mathcal{A}}Q^*_{\mathcal{F}}(\mathbf{s}',\mathbf{a}') - \max_{\mathbf{a}'\in supp(f_i)}Q^*_{\mathcal{F}}(\mathbf{s}',\mathbf{a}')\big)\big] \tag{50}$$

$$= \frac{\gamma}{1-\gamma}\max_{\mathbf{s},\mathbf{a}}\sum_{i=1}^{K}\alpha_i\mathbb{E}_{\mathbf{s}'}\big[\max_{\mathbf{a}'\in\mathcal{A}}Q^*_{\mathcal{F}}(\mathbf{s}',\mathbf{a}') - \max_{\mathbf{a}'\in supp(f_i)}Q^*_{\mathcal{F}}(\mathbf{s}',\mathbf{a}')\big] \tag{51}$$

$$= \frac{\gamma}{1-\gamma}\max_{\mathbf{s},\mathbf{a}}\sum_{i=1}^{K}\alpha_i\mathbb{E}_{\mathbf{s}'}\big[\Delta_{\mathcal{F},i}(\mathbf{s}')\big] \tag{52}$$

$$\leq \frac{\gamma}{1-\gamma}\max_{\mathbf{s}'}\sum_{i=1}^{K}\alpha_i\Delta_{\mathcal{F},i}(\mathbf{s}') \tag{53}$$

$$\leq \frac{\gamma}{1-\gamma}\sum_{i=1}^{K}\alpha_i\max_{\mathbf{s}'}\Delta_{\mathcal{F},i}(\mathbf{s}') \tag{54}$$

In step (46), from Lemma 1, we have $\mathcal{T}Q^*_{\mathcal{F}}(\mathbf{s},\mathbf{a}) \geq \mathcal{T}_{\mathcal{F}}Q^*_{\mathcal{F}}(\mathbf{s},\mathbf{a})$ $(\forall \mathbf{s},\mathbf{a})$.

Part 2: Proof of $\|Q^*_{\mathcal{F}} - Q^*\|_\infty \leq \frac{\gamma}{1-\gamma}\sum_{i=1}^{K}\alpha_i\max_{\mathbf{s}}\Delta'_{\mathcal{F},i}(\mathbf{s})$.

The proof follows a similar process as that of Part 1.

$$\|Q^*_{\mathcal{F}} - Q^*\|_\infty = \| \lim_{n\to\infty} (\mathcal{T}_{\mathcal{F}})^n Q^* - Q^* \|_\infty \tag{55}$$

$$= \| \lim_{n\to\infty} \big((\mathcal{T}_{\mathcal{F}})^n Q^* - Q^*\big) \|_\infty \tag{56}$$

$$= \| \lim_{n\to\infty} \sum_{k=0}^{n-1} \big((\mathcal{T}_{\mathcal{F}})^{k+1} Q^* - (\mathcal{T}_{\mathcal{F}})^k Q^*\big) \|_\infty \tag{57}$$

$$= \| \sum_{k=0}^{\infty} \big((\mathcal{T}_{\mathcal{F}})^{k+1} Q^* - (\mathcal{T}_{\mathcal{F}})^k Q^*\big) \|_\infty \tag{58}$$

$$\leq \sum_{k=0}^{\infty} \|(\mathcal{T}_{\mathcal{F}})^{k+1} Q^* - (\mathcal{T}_{\mathcal{F}})^k Q^* \|_\infty \tag{59}$$

$$\leq \sum_{k=0}^{\infty} \gamma^k \|\mathcal{T}_{\mathcal{F}} Q^* - Q^* \|_\infty \tag{60}$$

$$= \frac{1}{1-\gamma} \|\mathcal{T}_{\mathcal{F}} Q^* - Q^* \|_\infty \tag{61}$$

$$= \frac{1}{1-\gamma} \|\mathcal{T}_{\mathcal{F}} Q^* - \mathcal{T} Q^* \|_\infty \tag{62}$$

$$= \frac{1}{1-\gamma} \max_{\mathbf{s},\mathbf{a}} |\mathcal{T}_{\mathcal{F}} Q^*(\mathbf{s},\mathbf{a}) - \mathcal{T} Q^*(\mathbf{s},\mathbf{a})| \tag{63}$$

$$= \frac{1}{1-\gamma} \max_{\mathbf{s},\mathbf{a}} \big(\mathcal{T} Q^*(\mathbf{s},\mathbf{a}) - \mathcal{T}_{\mathcal{F}} Q^*(\mathbf{s},\mathbf{a})\big) \tag{64}$$

$$= \frac{1}{1-\gamma} \max_{\mathbf{s},\mathbf{a}} \big(\gamma \mathbb{E}_{\mathbf{s}'} \big[\max_{\mathbf{a}'\in\mathcal{A}} Q^*(\mathbf{s}',\mathbf{a}') \big] - \gamma \mathbb{E}_{\mathbf{s}'} \big[\sum_{i=1}^{K} \alpha_i \max_{\mathbf{a}'\in supp(f_i)} Q^*(\mathbf{s}',\mathbf{a}') \big] \big) \tag{65}$$

$$= \frac{\gamma}{1-\gamma} \max_{\mathbf{s},\mathbf{a}} \mathbb{E}_{\mathbf{s}'} \big[\max_{\mathbf{a}'\in\mathcal{A}} Q^*(\mathbf{s}',\mathbf{a}') - \sum_{i=1}^{K} \alpha_i \max_{\mathbf{a}'\in supp(f_i)} Q^*(\mathbf{s}',\mathbf{a}') \big] \tag{66}$$

$$= \frac{\gamma}{1-\gamma} \max_{\mathbf{s},\mathbf{a}} \mathbb{E}_{\mathbf{s}'} \big[\sum_{i=1}^{K} \alpha_i \max_{\mathbf{a}'\in\mathcal{A}} Q^*(\mathbf{s}',\mathbf{a}') - \sum_{i=1}^{K} \alpha_i \max_{\mathbf{a}'\in supp(f_i)} Q^*(\mathbf{s}',\mathbf{a}') \big] \tag{67}$$

$$= \frac{\gamma}{1-\gamma} \max_{\mathbf{s},\mathbf{a}} \mathbb{E}_{\mathbf{s}'} \big[\sum_{i=1}^{K} \alpha_i \big(\max_{\mathbf{a}'\in\mathcal{A}} Q^*(\mathbf{s}',\mathbf{a}') - \max_{\mathbf{a}'\in supp(f_i)} Q^*(\mathbf{s}',\mathbf{a}') \big) \big] \tag{68}$$

$$= \frac{\gamma}{1-\gamma} \max_{\mathbf{s},\mathbf{a}} \sum_{i=1}^{K} \alpha_i \mathbb{E}_{\mathbf{s}'} \big[\max_{\mathbf{a}'\in\mathcal{A}} Q^*(\mathbf{s}',\mathbf{a}') - \max_{\mathbf{a}'\in supp(f_i)} Q^*(\mathbf{s}',\mathbf{a}') \big] \tag{69}$$

$$= \frac{\gamma}{1-\gamma} \max_{\mathbf{s},\mathbf{a}} \sum_{i=1}^{K} \alpha_i \mathbb{E}_{\mathbf{s}'} \big[\Delta'_{\mathcal{F},i}(\mathbf{s}') \big] \tag{70}$$

$$\leq \frac{\gamma}{1-\gamma} \max_{\mathbf{s}'} \sum_{i=1}^{K} \alpha_i \Delta'_{\mathcal{F},i}(\mathbf{s}') \tag{71}$$

$$\leq \frac{\gamma}{1-\gamma} \sum_{i=1}^{K} \alpha_i \max_{\mathbf{s}'} \Delta'_{\mathcal{F},i}(\mathbf{s}') \tag{72}$$

Therefore, by combining Part 1 and 2, we get

$$\|Q^*_{\mathcal{F}} - Q^*\|_\infty \leq \frac{\gamma}{1-\gamma} \min \left(\sum_{i=1}^{K} \alpha_i \max_{\mathbf{s}} \Delta_{\mathcal{F},i}(\mathbf{s}), \sum_{i=1}^{K} \alpha_i \max_{\mathbf{s}} \Delta'_{\mathcal{F},i}(\mathbf{s}) \right)$$

□

A.6 Proof of Theorem 5

Proof. This theorem has a similar form as that of Theorem B.1 of [21], except that the operator analyzed in this paper is $\mathcal{T}_{\mathcal{F}}$. The proof also follows a similar process. Specifically, $\forall n \in \mathbb{N}$,

$$\|\widehat{Q}_{\mathcal{F}}^{(n+1)} - Q_{\mathcal{F}}^*\|_\infty = \|\widehat{Q}_{\mathcal{F}}^{(n+1)} - \mathcal{T}_{\mathcal{F}}\widehat{Q}_{\mathcal{F}}^{(n)} + \mathcal{T}_{\mathcal{F}}\widehat{Q}_{\mathcal{F}}^{(n)} - Q_{\mathcal{F}}^*\|_\infty \tag{73}$$

$$\leq \|\widehat{Q}_{\mathcal{F}}^{(n+1)} - \mathcal{T}_{\mathcal{F}}\widehat{Q}_{\mathcal{F}}^{(n)}\|_\infty + \|\mathcal{T}_{\mathcal{F}}\widehat{Q}_{\mathcal{F}}^{(n)} - Q_{\mathcal{F}}^*\|_\infty \tag{74}$$

$$\leq \delta + \|\mathcal{T}_{\mathcal{F}}\widehat{Q}_{\mathcal{F}}^{(n)} - Q_{\mathcal{F}}^*\|_\infty \tag{75}$$

$$= \delta + \|\mathcal{T}_{\mathcal{F}}\widehat{Q}_{\mathcal{F}}^{(n)} - \mathcal{T}_{\mathcal{F}}Q_{\mathcal{F}}^*\|_\infty \tag{76}$$

$$\leq \delta + \gamma\|\widehat{Q}_{\mathcal{F}}^{(n)} - Q_{\mathcal{F}}^*\|_\infty \tag{77}$$

Now we prove that

$$\|\widehat{Q}_{\mathcal{F}}^{(n+1)} - Q_{\mathcal{F}}^*\|_\infty \leq \sum_{k=0}^{n} \delta\gamma^k + \gamma^{n+1}\|\widehat{Q}_{\mathcal{F}}^{(0)} - Q_{\mathcal{F}}^*\|_\infty. \tag{78}$$

From (77), let $n = 0$, we get the base case

$$\|\widehat{Q}_{\mathcal{F}}^{(1)} - Q_{\mathcal{F}}^*\|_\infty \leq \delta + \gamma\|\widehat{Q}_{\mathcal{F}}^{(0)} - Q_{\mathcal{F}}^*\|_\infty. \tag{79}$$

If (78) holds for $n = n'$, and from (77),

$$\|\widehat{Q}_{\mathcal{F}}^{(n'+1+1)} - Q_{\mathcal{F}}^*\|_\infty \leq \delta + \gamma\|\widehat{Q}_{\mathcal{F}}^{(n'+1)} - Q_{\mathcal{F}}^*\|_\infty \tag{80}$$

$$\leq \delta + \gamma\left(\sum_{k=0}^{n'} \delta\gamma^k + \gamma^{n'+1}\|\widehat{Q}_{\mathcal{F}}^{(0)} - Q_{\mathcal{F}}^*\|_\infty\right) \tag{81}$$

$$\leq \sum_{k=0}^{n'+1} \delta\gamma^k + \gamma^{(n'+1)+1}\|\widehat{Q}_{\mathcal{F}}^{(0)} - Q_{\mathcal{F}}^*\|_\infty \tag{82}$$

Thus, (78) also holds for $n = n' + 1$. Combining this with the base case (79), we have that (78) holds for all $n \in \mathbb{N}$.

Then, let $n \to \infty$ on both sides of (78), we have

$$\lim_{n\to\infty} \|\widehat{Q}_{\mathcal{F}}^{(n)} - Q_{\mathcal{F}}^*\|_\infty \leq \frac{\delta}{1-\gamma}. \tag{83}$$

□

B Parameters in Case Study

We set equal weight to each piece of the knowledge, specifically, in Table 1.

Table 2: Parameter setting in domain knowledge

	Experiment name	$\boldsymbol{\alpha}$
1	Ten recently interacted nodes	$[1, 0, 0, 0]$
2	Around recently interacted nodes	$[0, 1, 0, 0]$
3	Three generations around me-person	$[0, 0, 1, 0]$
4	Direct relatives	$[0, 0, 0, 1]$
5	Ten recently interacted nodes OR Around recently interacted nodes	$[0.5, 0.5, 0, 0]$
6	Ten recently interacted nodes OR Three generations around me-person	$[0.5, 0, 0.5, 0]$
7	Ten recently interacted nodes OR Direct relatives	$[0.5, 0, 0, 0.5]$
8	Around recently interacted nodes OR Three generations around me-person	$[0, 0.5, 0.5, 0]$
9	Around recently interacted nodes OR Direct relatives	$[0, 0.5, 0, 0.5]$
10	Three generations around me-person OR Direct relatives	$[0, 0, 0.5, 0.5]$
11	Ten recently interacted nodes OR Around recently interacted nodes OR Direct relatives	$[0.33, 0.33, 0, 0.33]$
12	All nodes	N/A
13	Last interacted node (baseline)	N/A