# Allowing for the Grounded Use of Temporal Difference Learning in Large Ranking Models via Substate Updates

Daniel Cohen*
daniel_cohen@brown.edu
Brown University
Providence, RI

## ABSTRACT

We introduce a modification of an established reinforcement learning method to facilitate the widespread use of temporal difference learning for IR: interpolated substate temporal difference (ISSTD) learning. While reinforcement learning methods have shown success in document ranking, these contributions have relied on relatively antiquated policy gradient methods like REINFORCE. These methods bring associated issues like high variance gradient estimates and sample inefficiency, which presents significant obstacles when training deep neural retrieval models. Within the reinforcement learning community, there exists a substantial body of work on alternative methods of training which revolve around temporal difference updates, such as Q-learning, Actor-Critic, or SARSA, that resolve some of the issues seen in REINFORCE. However, temporal difference methods require the full size of the state to be modeled internally within the ranking model, which is unrealistic for deep full text retrieval or first stage retrieval. We therefore propose ISTD, operating on the substate, or individual documents in the case of matching models, and interpolating the temporal difference updates to the rest of the state. We provide theoretical guarantees on convergence, enabling the drop in use of ISSTD for any algorithm that relies on temporal difference updates. Furthermore, empirical results demonstrate the robustness of this approach for deep neural models, outperforming the current policy gradient approach for training deep neural retrieval models.

## CCS CONCEPTS

• **Information systems → Retrieval models and ranking**; • **Computing methodologies → Temporal difference learning**.

## KEYWORDS

information retrieval, reinforcement learning, temporal difference, ranking

**ACM Reference Format:**
Daniel Cohen. 2021. Allowing for the Grounded Use of Temporal Difference Learning in Large Ranking Models via Substate Updates. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '21), July 11–15, 2021, Virtual Event, Canada.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3404835.3462952

*Work done while at University of Massachusetts Amherst

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
*SIGIR '21, July 11–15, 2021, Virtual Event, Canada*
© 2021 Association for Computing Machinery.
ACM ISBN 978-1-4503-8037-9/21/07…$15.00
https://doi.org/10.1145/3404835.3462952

## 1 INTRODUCTION

The core objective of an information retrieval (IR) model is to identify the documents or passages in a collection that sufficiently address the information needs of the user. As there can be multiple items that accomplish this, the objective of search in IR is then to rank a collection under some external metric, be it to satisfy the user, ensure diversity, or maintain fairness [2, 13, 26]. Recently, reinforcement learning (RL) methods have been introduced as alternative methods to achieve effective performance [14, 21, 34, 42]. Under this regime, retrieval models (policies) can directly maximize any reward without relying on fully or weakly supervised labels. Almost all of these works rely on one specific method in RL when used for IR: policy gradient via REINFORCE [34, 35, 37, 39]. REINFORCE based retrieval models treat the documents as a learned probability distribution and iteratively samples them to determine a ranking. After all documents have been ranked, the model is updated such that it will adjust this distribution to increase the chance of sampling relevant documents based on the rewards it received. An additional advantage and a powerful property of this approach beyond directly maximizing some non-traditional, non-convex reward function is the ease of incorporating this approach into any model over any input size. A model trained via REINFORCE can score documents individually with the only requirement being to model all choices in a distribution via a softmax operation. In this way, the model can incorporate information from all candidate non-relevant documents when updating its weights. However, policy gradient and REINFORCE methods suffer from significant drawbacks. First, updates to the model only occur at the end of a session or episode. This results in online updating becoming a challenge. Second, the gradient approximation has very high variance as document rankings are selected via Monte Carlo sampling [28].

To remedy these shortcomings, other works have proposed complementary methods or alternative approaches [1, 10–12, 33]. The core of these approaches rely on *temporal difference* (TD) updates. Rather than only updating the model at the very end of an episode, TD methods incrementally update the model after each document ranking. Unfortunately, TD updates do not rely on a softmax operation and are thus unusable when the input is very large, such as ranking the top $n$ documents in neural architectures [8, 23].

In this paper, we show a modification of TD based algorithms, Interpolated Sub-State TD (ISSTD), which allows current IR models to be directly trained via TD updates without any modifications, acting as a "drop-in" optimization method similar to REINFORCE based approaches for IR tasks. This result provides the necessary machinery for TD based methods like Q-learning and Actor-Critic to be used in any ranking task where the only current option is REINFORCE. Theoretical analysis bounds the error introduced in this method as well as convergence guarantees when certain reasonable conditions are met. Empirically, we demonstrate that ISSTD methods are effective for ranking and leverage our theoretical result into novel RL methods to outperform the current REINFORCE approach for a number of neural architectures. Furthermore, ISSTD methods can achieve close to supervised performance, which represents an approximate upper bound of the achievable performance of a model trained using RL [28]. With these two results, our proposed approach enables the theoretically grounded use of recent RL research in any IR task and for any architecture, regardless of the size of the input.

## 2 BACKGROUND

### 2.1 Reinforcement Learning

As this paper relies heavily on a theoretical understanding of RL to extend TD methods to IR, we present the necessary background. The core premise of RL is that it presents a framework for an agent to learn from its actions within an environment. This can range from canonical examples like robotic control [38] and game playing [7] to even machine translation [36]. The agent can be a neural model, child, program, or any process that is able to improve its future actions based on the input it receives from the environment. In the case of game playing, the environment would be a video game while for machine translation it would be represented as the space of possible translations and training examples. This environment partially determines the reward to convey how well an agent's decisions satisfy some unknown objective. In our case of IR, the environment is the set of documents to be ranked and the input signal would be a metric used to model user satisfaction or relevance, such as session duration, nDCG, MAP, etc. The last core principle of RL is the sequential nature of decision-making.

A critical step in representing any task as a RL problem is formally defining a Markov decision process (MDP). This MDP is a mathematical model that determines the environment which includes what we want our agent to learn. Defined as the tuple $(\mathcal{S}, \mathcal{A}, P, R, d_0, \gamma)$ representing the state space, action space, transition function $P : \mathcal{S}\times\mathcal{A}\times\mathcal{S} \to [0, 1]$, reward function $R : \mathcal{S}\times\mathcal{A} \to \mathbb{R}$, initial state distribution $d_0$ and decay parameter $\gamma$.

The objective is then to find a policy $\pi : \mathcal{S} \to \mathcal{A}$ that maximizes the expected total discounted reward $R_t \sim r(s_t, a_t)$ that the agent can obtain,

$$J(\pi) = \mathbb{E}\Big[\sum_{t=0}^{\infty} \gamma^t R_t \Big| \pi\Big].$$

Closely related to this goal is the value function,

$$V^{\pi}(s) = \mathbb{E}\Big[\sum_{k=0}^{\infty} \gamma^t R_{t+k} \Big| s_t = s, \pi\Big] = \mathbb{E}\Big[G_t \Big| s_t = s, \pi\Big] \tag{1}$$

which is the expected discounted return if the agent follows policy $\pi$ from state $s$. This differs from $J$ as the discounted return is now conditioned on a specific state rather than over all possible situations. The goal is then to find some $\pi^*$ that achieves the highest possible value for each state:

$$V^*(s) = \max_{\pi\in\Pi} \mathbb{E}\Big[G_t \Big| s_t = s, \pi\Big] \ \forall s \in \mathcal{S}.$$

The parallel to IR can be viewed as the value function measuring how well our retrieval model $\pi$ can rank a set of documents. We would ideally want a retrieval model that can maximize our value function for every query and set of candidate documents.

There exists a large body of work on finding a $\pi^*$, and this paper will focus on TD learning for IR as an alternative to approaches relying on Markov chain Monte Carlo sampling methods such as policy gradient that have already been adapted for retrieval tasks [31, 34, 37, 39, 41]. The fundamental idea behind TD learning is to bootstrap from some initial value estimate via the Bellman equation:

$$TV(s) = r(s, a) + \gamma\mathbb{E}[V(s')] \tag{2}$$

where $s'$ is the next state visited and will converge under certain conditions [28]. It is straightforward to see where the concept of TD comes from, $T$ updates $V$ based on the expected next state's discounted cumulative reward. Q-learning, SARSA, and other TD based methods update via the function $Q : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ shown below

$$Q(s, a) = \mathbb{E}\Big[G_t \Big| s_t = s, a_t = a, \pi\Big] \tag{3}$$

under the same contraction operator, $T$. While $V$ represents the cumulative reward from state $s$, $Q$ captures the cumulative reward from state $s$ having taken action $a$.

Here, we highlight a key difference between $Q$-learning and SARSA: the update for $Q$-learning will always be greedy, meaning that the update under $T$ will take $Q$ value of whatever action is best in the next state $s'$. This method of updating is considered off-policy, as the agent is selecting actions that the current policy might not necessarily take. In contrast, SARSA will follow the underlying $\pi$ and often takes an $\epsilon$-greedy approach when selecting the next state to update via TD learning. This off-policy approach suffers from divergence in conditions where SARSA will converge [28]. This plays an integral role for ISSTD later in this work.

Lastly, there are no limits on what defines $Q$ or $V$ in terms of a function. These can be modeled via a table and trained under dynamic programming, a linear function approximator $Q(s, a) = \theta^{\top}\phi(s)$ where $\phi$ is some state representation or a nonlinear function approximator like a neural net. Although all nonlinear function approximations of $Q$ or $V$ discard all convergence guarantees; nonetheless, these nonlinear approaches achieve remarkable results in challenging environments [4, 7, 38].

### 2.2 Related Work

The concept of directly maximizing some reward modeled by a traditional IR metric has been well explored within the community. The majority of these approaches, and all of those used for ranking a large set of documents, rely on policy gradient and the REINFORCE algorithm. For learning to rank tasks, Wei et al. introduce MDPRank [34] which treats each query and a list of documents as a single episode, and each action selects the most relevant document of the yet to be ranked documents. They achieve competitive results using REINFORCE to directly optimize the ranker. Using the same MDP formulation and REINFORCE, MDPDIV [37] uses an alternative reward signal to maximize diversity within a ranked list, demonstrating the flexibility of an RL based optimization. Nogueira and Cho [20] use this same optimization approach to reformulate a query for improved retrieval performance, and while not the core contribution of their work, Wang et al. [31] use REINFORCE to update the generator in the IRGAN framework due to the magnitude of the state space. Zeng et al. [41] introduce a policy optimized via REINFORCE that directly attempts to improve multi-page search via treating each page generation as a single step in an MDP. Recently, Xu et al. [39] expand on MDPRank [34] by introducing a pairwise policy gradient approach to reduce the variance found in the Markov chain Monte Carlo method of sampling in REINFORCE. At each step, the policy gradient agent observes the potential reward of taking two actions and updates the weights of the policy accordingly. They demonstrate both theoretically and empirically the impact of reducing the variance of the gradient estimate within the IR environment and motivate the need to incorporate modern RL methods for IR tasks. We have not observed any previous work investigating the effective use of REINFORCE for deep neural retrieval beyond a two-layer feedforward network.

However, there has been significant work that leverages potentially more capable methods than REINFORCE for related tasks where we do not observe the limitations discussed above. In the case of item recommendation, Liu et al. [15] use a deep actor-critic framework that relies on a TD component to reduce the variance of policy gradient for item recommendation with marked success. They also incorporate a replay buffer, a method that results in significant improvement when incorporating TD learning [7]. Their method represents the objective of our contribution, as enabling efficient TD learning for very large state spaces will facilitate the use of their more sophisticated methods such as actor-critic and prioritized memory replay for document and passage ranking. Wang and Jin [32] use the same actor-critic setup for multi-step coarse to fine question answering, where the largest state in the MDP is a single document.

Hu et al. [9] incorporate a variation of actor-critic called deep deterministic policy gradient for the task of e-commerce sessions. This approach further reduces the variance by allowing the agent to operate only over the state space [12]. In addition, through a novel construction of an e-commerce session MDP, the critic component that relies on TD learning is able to perform a full backup.

To the best of our knowledge, there exists no work using TD methods for IR where the state, or set of candidate documents, is too large to compute in a single model. The reason for this is a symptom of the Bellman update $T$ and is discussed in detail in Section 4.

## 3 MARKOV DECISION PROCESS

MDP construction directly impacts what the agent learns and is a non-trivial task when translating real-world environments (IR) into this formulation. As the contribution of the paper is the ISS based agent for any MDP, we adopt the well-defined MDP proposed by Wei et al. for feature-based L2R as the structure for the environment as it favors the REINFORCE baseline while also empirically shown to be a stable setting for RL retrieval models [34].

**State:** We construct the state $s \in \mathcal{S}$ as

$$[q, D_{ur}, t] \tag{4}$$

| Symbol | Definition |
|---|---|
| $\mathcal{S}$ | State space |
| $s \in \mathcal{S}$ | State |
| $z \in s$ | Substate |
| $\mathcal{A}$ | Action space |
| $a \in \mathcal{A}$ | Action |
| $\pi$ | Policy |
| $f_{IR}$ | Retrieval model |
| $\theta$ | Parameters |
| $a', s', \theta'$ | Action, State,$\theta$ at next step |
| $t$ | Time step |
| $T : TV(s) = r(s,a) + \gamma\mathbb{E}[V(s')]$ | Bellman Update |
| $Q : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ | Q-function |
| $V : \mathcal{S} \to \mathbb{R}$ | Value function |
| $B(\mathcal{X})$ | Functional space over some $\mathcal{X}$ space |
| $\mathcal{P} : B(\mathcal{X}) \to \mathbb{R}^b$ | Evaluation Operator |
| $F : \mathbb{R}^b \to B(\mathcal{X})$ | Interpolate Operator |
| $\gamma$ | Discount rate |
| $\alpha$ | Learning rate |
| $\hat{Q}, \hat{V}, \hat{\theta}$ | Converged fixed point |
| $Q^*, V^*, \theta^*$ | Optimal points |

**Table 1: Table of definitions.**

where $q$ is a query, $D_{ur}$ is a list of unranked documents where $d_i \in D_{ur} = [d_1, d_2, ..., d_n]$, and $t$ represents the current time step in an episode. The terminal state occurs when $|D_{ur}| = 1$. A substate $z_i$ can then be viewed as $[q, d_i, t]$ where $d_i \in D_{ur}$. In the case of deep learning, both $q$ and $d_i$ are represented as sequences of words $w$ from a shared vocabulary: $q = \{w_i\}_1^m,\ d = \{w_i\}_1^n$ for a query of length $m$ and a document of length $n$.

**Action:** At each step, the policy chooses a document to rank next from the set of unranked documented, $D_{ur}$. Each candidate represents an element in $\mathcal{A}$.

**Reward:** We use reciprocal rank below, where $D_r$ is the set of all relevant documents with respect to $q$ and $D_{nr}$ are all other non-relevant documents. The reward function $r(s, a)$ is formally defined as

$$r(s,a) = \begin{cases} 0 & a \in D_{nr} \\ \frac{1}{t} & a \in D_r \end{cases}$$

with $a$ representing the selection of a single document.

**Transition:** The transition function $\mathcal{T} : \mathcal{S} \times \mathcal{A} \to S$ models the dynamics of the environment and maps a state, action pair to a new state. We structure the transition as a deterministic function,

$$\mathcal{T}(s_t, a_t) = [q, D_{ur} \setminus d_{a_t}, t + 1],$$

where $d_{a_t}$ represents the document chosen at step $t$ by action $a$. We set $D_{ur}$ in $d_0$ to be $n$ highest scoring non-relevant documents from the collection under BM25 [25] with $q$. We select $n = 10$ for ease of analysis in our evaluation of our findings.

## 4 REINFORCEMENT LEARNING FOR RETRIEVAL

In this section, we first introduce how RL is used for ranking, and then we discuss why TD methods fail in the current IR regime. We will then introduce ISSTD to gracefully handle these conditions. A typical ranking model scores each document independently and produces a ranking from these separate scores. Thus, one can view individual documents $d_i$ of a collection as a sub-state $z_i \in s$. In this representation, $Z$ creates a partition of $s$ such that $\bigcup z_i = s$, $\bigcap z_i = \varnothing$. However, a state $s \in \mathcal{S}$ consists of all documents to be ranked for a query or session. As discussed, policy gradient based methods are a drop-in optimization method for ranking models. This convenience is possible, as despite each document being scored independently, policy gradient methods create a distribution over all possible documents via the softmax function when selecting the next document:

$$\pi(s) = \frac{\exp(f_{IR}(q, d_i))}{\sum_{j \in |D_{ur}|} \exp(f_{IR}(q, d_j))} = \frac{\exp(f_{IR}(z_i))}{\sum_{j \in |Z|} \exp(f_{IR}(z_j))}.$$

This representation enables independent document ranking regardless of the amount of candidate documents in $s$. In the case of TD learning, this is not necessarily true. For example, in Q-learning [33] where the TD update occurs via

$$Q(s, a) = Q(s, a) + \alpha(s, a)(r(s, a) + \gamma \max_{a'} Q(s', a') - Q(s, a)), \quad (5)$$

the update, in either table lookup, linear, or nonlinear function approximation, is reducing the error for the entire state's representation. However, in the case of IR situations, we are unable to model a single $s$ entirely, and substitute $Q(s, a)$ on the right hand side of Eq.5 for the IR model operating over a single document. Thus the actual update that would occur is

$$Q(s, a) = f_{IR}(z) + \alpha(s, a)(r(s, a) + \gamma \max_{z'} f_{IR}(z') - f_{IR}(z)). \quad (6)$$

This now computes the value of a substate trajectory through the MDP, removing information about all documents when updating the IR model except for the documents $z$ and $z'$, which are *substates* of $s$. In order for TD to be used in IR, it would require a significantly large compute cluster to handle all the documents in the entire state simultaneously to allow for the use of Eq. 5. In the case of deep learning, this is simply not feasible. Furthermore, competitive TD methods rely on large memory replay buffers, which compounds this issue due to the need to recompute $2|b_m||s|$ documents for each update where $|b_m|$ is the batch size of the buffer. In the remainder of this section, we introduce machinery such that Eq. 6, with a small modification, is a valid substitute for Eq. 5 without loss of guarantees. This modification facilitates the drop in use for any typical IR task by showing that a standard learning to rank process is a special case of an interpolated value function maintaining convergence properties. Specifically, we show why Eq. 6 is a grounded RL method for ranking situations where the model operates on individual documents.

## 4.1 Interpolated Sub-State Temporal Difference Learning

The key result in this section is that in function approximation cases, where we rely on some parameterization $\theta$ to determine a ranking, retrieval models trained using ISSTD over individual documents will converge to a similar policy as the one trained with TD methods operating on the entire document list. This parallel is showcased in Figure 1, where we observe the substate trajectory as individual documents.

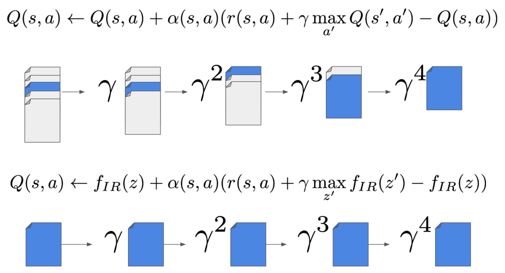
$Q(s,a) \leftarrow Q(s,a) + \alpha(s,a)(r(s,a) + \gamma \max_{a'} Q(s',a') - Q(s,a))$
$\gamma$
$\gamma^2$
$\gamma^3$
$\gamma^4$
$Q(s,a) \leftarrow f_{IR}(z) + \alpha(s,a)(r(s,a) + \gamma \max_{z'} f_{IR}(z') - f_{IR}(z))$
$\gamma$
$\gamma^2$
$\gamma^3$
$\gamma^4$

**Figure 1: Demonstration of the ISSTD regime used for TD agents on IR tasks (bottom) compared to standard TD learning (top). As the reward is determined solely on the relevant document, the final return is equivalent.**

To accomplish this, we will first introduce the update rule for $\theta$ (Eq. 7) and then provide the machinery needed to share update information across separate points. Next, we show that the conventional ranking method used in Eq. 6 satisfies the necessary requirements for a valid interpolation operation across a set of basis points in Lemma 4.1. Having established this result, we then proceed to draw a parallel to previous work in standard state interpolation to prove the convergence of Eq. 6. Finally, we connect the substate view of Eq 6 to the full state view of Eq. 5 in Corollary 4.2.1 to confirm the similarity of their converged policies.

In addition, we wish to highlight the limitation of the following work as it assumes a direct mapping between a substate and the reward. If the reward function depends on more than one document per step such that Figure 1 is no longer an accurate depiction, then the requirements on the interpolation are violated due to the loss of the nonexpansion property.

*4.1.1 Updating $\theta$:* Consider the case where we are using function approximation for $Q$, parameterized by $\theta \in \mathbb{R}^b$. As we care about the updates to our model, we will be focusing on the functional space of $Q_\theta : B(\mathcal{S} \times \mathcal{A}) \to \mathbb{R}$. A TD update for each dimension of $\theta$ under the Bellman operator ($T$) from Eq. 2 can be done via

$$\Delta\theta_{ti} = \alpha_{ti}\beta(z_i, a_i, s)(R_t + \gamma \max_{a'} Q_{\theta_t}(s', a') - \theta_{ti}), \quad (7)$$

where $z_i \in Z$ is a set of basis points of $s$, $Q_{\theta_t}$ is an interpolator over $S$ (such as the $Q$ or $V$ function defined in Eq. 1 and 5), $\alpha_{ti}$ is the learning rate for dimension $i$ of $\theta$ at time $t$, and $\beta$ is a bounded measurable smoothing function [29]. A key note in this formulation is that $\beta$ allows the potential updating of multiple components of $\theta_t$ for each basis point $z_i$, allowing for a single update to affect the value estimates of other states. Without loss of generality, the value function and action-value function can be used interchangeably in this section [29].

*4.1.2 Decomposing and Interpolating:* We next introduce two definitions, a decomposition method and an interpolator, necessary to show equivalence with Eq. 6 and enabling the full use of TD methods for ranking large states. Intuitively, the operator $\mathcal{P}$ decomposes the value function to only operating on the basis points, and then $F$ interpolates its values across the parameter space.

**Definition 4.1.** $\mathcal{P} : B(\mathcal{S}) \rightarrow \mathbb{R}^b$ is a composite pointwise evaluation operator with respect to a fixed set of basis points $Z = \{(z_1, v_1), \ldots, (z_n, v_n)\}$ if $(\mathcal{P}V)_i = V(z_i)$.

**Definition 4.2.** Let $F : \mathbb{R} \rightarrow B(\mathcal{S})$ be mapping from parameters to functions over the space $\mathcal{S}$. Then $F$ is interpolative with respect to the set of basis points of $\mathcal{S}$ if for all $V \in B(\mathcal{S})$, $\mathcal{P}F\mathcal{P} = \mathcal{P}$. Furthermore, $F$ is non-expansive interpolator if for the set of basis points $Z$ and corresponding values $H = \{h_1, \ldots, h_n\}$, then for any parameterization $\theta$, $F(\theta)(z_i) = h_i$.

The nonexpansive property of $F$ requires that the evaluation of the value function at the basis points does not change during this interpolation to preserve the function when reconstructed. Having these properties results in $F\mathcal{P}$ acting as an interpolative nonexpansion, which implies $V_{t+1} = F\mathcal{P}TV_t$ converges to $V^*$ despite the nontraditional value updates [29]. Using this established result, we show that the conventional pointwise evaluation used in ranking models is an extension of the above.

To satisfy this objective, we introduce the idea that the common pointwise scoring used in ranking models is a valid interpolation. A close inspection of Eq. 6 shows that there is no smoothing being done when updating the $Q$ function in this setting. This operation can therefore be modeled as a first-order spline interpolation $F$ over $\theta$, visually represented in Figure 2. In the case of ranking, each document represents a basis point of our state space, and $F$ creates a Voronoi diagram where any point in the space is assigned the value of the document that is closest to it. We provide the formal discussion below:

*Lemma* 4.1. The first order spline interpolation $F$, $Fu = \sum_{i=1}^{b} u_i \mathbf{1}_{A_i}$ is a measurable non-expansion in the sup-norm over some basis set $Z$ and $\mathbf{1}_{A_i}$ as the characteristic function of A such that

$$\mathbf{1}_{A_i}(x) = \begin{cases} 1 & x \in A_i \\ 0 & x \notin A_i \end{cases}$$

Proof. We define $A_i$ to be a partition of $Z$ such that $A_i$ covers the $k$-nearest neighborhood around $z_i$ with $k = 1$. This results in a piecewise continuous evaluation over basis $Z$ where $Fu_i = u_i$ for the entire space $A_i$ around point $z_i$. Then we observe $||Fu||_\infty = ||u||_\infty$. For any $u, g$ in the same Banach space,

$$||Fu - Fg||_\infty = ||u - g||_\infty \tag{8}$$

which satisfies the requirement of a non-expansion. $F$ is also piecewise continuous and measurable, i.e. $\sum_{i=1}^{b} u_i \mathbf{1}_{A_i}$ is also measurable. □

This spline interpolation, visually represented in Figure 2, acts as our smoothing function for updates across our basis points $Z$.

*4.1.3 ISSTD Convergence:* Now that we have defined a smoothing function that satisfies Eq. 6 used for ranking, we next show that it will converge under certain conditions by leveraging a previous result which shows convergence when updates are shared across entirely different states [29]:

Proposition 4.2. *Let $V : B(\mathcal{S}) \rightarrow \mathbb{R}$ be a linear function approximator defined by Eq. 1, then the substate update*

$$V(s) = f_{IR}(z) + \alpha(s)(r(s,a) + \gamma\mathbb{E}[f_{IR}(z')] - f_{IR}(z)) \tag{9}$$

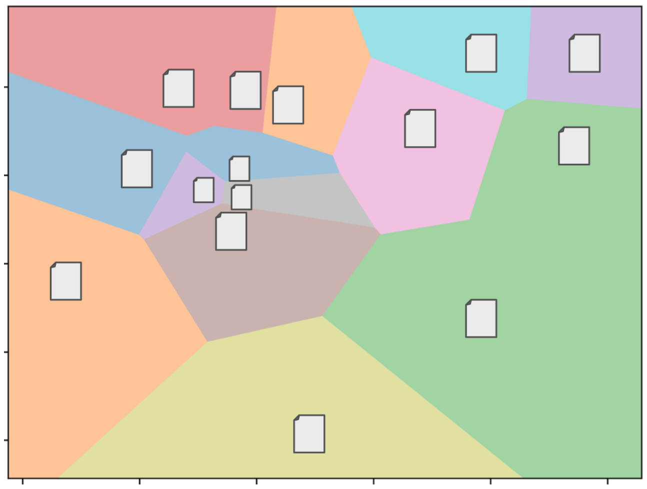

**Figure 2: A visual representation of how substate values partition the state space. The documents represent the basis points and the first order spline operation interpolates their values to their surrounding neighborhoods.**

*converges to a $\hat{V}^*$ if (i) $\mathcal{P}V$ exists, (ii) $F$ is a non-expansive interpolation such that $||Ff_1 - Ff_2|| \leq \gamma||f_1 - f_2||$, and (iii) $Z = \{(z_i, v_i)\}_1^n$ is the set of basis points of $s$ such that $Z$ creates a partition of $s$ with $\bigcup z_i = s$, $\bigcap z_i = \varnothing$.*

Proof. In order to do so, we use the algorithm introduced by Gordon [5], which includes the projection and interpolation operators previously defined, to update the weights of the value function from Eq. 1,

$$\theta_{t+1} = \mathcal{P}TF\theta_t \tag{10}$$

where $T$ is the Bellman operator.

Then we can see that value iteration can be modeled via

$$V_{t+1} = TF\mathcal{P}V_t. \tag{11}$$

This provides the critical first step of the drop-in placement of TD updates for training ranking models. In this representation, $\mathcal{P}$ acts as a decomposition of $V$ to the point-wise evaluation of the expected return of that basis point. For ranking, these are individual document scores over the entire state. As $F$ is a non-expansion, then $F\theta_t$ converges to $\hat{\theta}^*$ [5]. We observe a direct mapping of a satisfactory $\mathcal{P}$ to the case seen in Eq. 9 where $f_{IR}$ is a composite pointwise evaluation of $V$ such that each basis point consists of $d_i$. Formally,

$$\theta_{IR}(z_i) = (\mathcal{P}V)_i \tag{12}$$

deconstructs some larger retrieval model that fully captures all possible documents within a reranking list or a collection as its input and decomposes it to individual functions over each $z_i$.

Next, we select an $F$ that can interpolate the mapping from the parameter $\theta$ to a $V \in B(\mathcal{S})$ with respect to $S$. Szepesvári and Smart [29] prove that if $F\mathcal{P}$ is a interpolative non-expansion, then $V_{t+1} = F\mathcal{P}TV_t$ converges to $V^*$. By Lemma 4.1, a first order spline is a non-expansion. As the smoothing is 0 everywhere except where the characteristic function is active, $\beta(z_i, a, s)$ in Eq. 7 becomes an indicator variable. This results in $\beta_{ti} = 0$ for everywhere except for basis $z_i$.

Thus, as long as

$$\sum_{i=0}^{\infty} \alpha = \infty, \sum_{i=0}^{\infty} \alpha^2 < \infty \quad (13)$$

meaning all states are visited infinitely often, then $TV(z) \rightarrow \hat{V}^*$ and $TQ \rightarrow \hat{Q}^*$ even when individual substates are updated independently for value iteration and Q-learning respectively [29]. □

While we show that typical learning to rank updates are a special case of interpolation, observe that more suitable smoothing functions such as radial basis function kernels or Gaussian processes satisfy the nonexpansive properties of $F$ while potentially improving sample efficiency. We leave the evaluation of alternative smoothing methods to future work.

*4.1.4 Similarity of ISSTD and the true Q-function:* Lastly, we show that the $Q$-function learned via ISSTD approximates the true $Q$-function if the retrieval model were to observe the entire document ranked list.

*Corollary* 4.2.1. Let $|| \max_z f_{IR}(z) - Q(s,a)|| \leq \epsilon$ such that

$$\mathbb{E}[|| \max_z f_{IR}(z) - Q(s,a)||] = 0$$

under any MDP such that every state is visitable, $\lim_{t\to\infty} P(s_t = s_{\text{terminal}}) = 1$, and $\sum_{i=0}^{\infty} \alpha = \infty$, $\sum_{i=0}^{\infty} \alpha^2 < \infty$, then

$$Q_{ISS}(s,a) = f_{IR}(z) + \alpha(s,a)(r(s,a) + \gamma \max_{z'} f_{IR}(z') - f_{IR}(z)) \quad (14)$$

converges to oscillate within a region $C$ of the fixed a point $\hat{Q}^*$ defined by

$$Q(s,a) = Q(s,a) + \alpha(s,a)(r(s,a) + \gamma \max_{a'} Q(s',a') - Q(s,a)) \quad (15)$$

Proof. If we treat a $\pi$ that operates on an entire $s$ as an epsilon greedy $\pi(s) = \text{argmax}_a Q(s,a)$, then $Q_{ISS}$ follows the exact policy as the underlying generating policy from Q and becomes SARSA in this instance. We then represent $Q_{ISS}$ to be a linear function approximator of $Q$. If we define $Q(s,\cdot)$ as

$$Q(s,\cdot) = < f_{IR}(z_1), \ldots, f_{IR}(z_n) >, \quad (16)$$

then we observe that $Q_{ISS}$ is the operation $\theta_{ISS}{}^{\intercal} Q(s,\cdot)$ and acts as one hot vector such that $||\theta_{ISS}||_\infty = 1, ||\theta_{ISS}||_2 = 1$. Then this is a linear function approximator of the $\pi$ generated by following $Q$-learning, i.e. SARSA. We leverage the result from Gordon [6] that states that SARSA will converge to linear region around the generating $\pi$ under linear function approximation. Thus $Q_{ISS}$ will converge to a region $C$ around the fixed point $\hat{Q}^*$. □

Having established that operating on individual documents is a valid extension of TD methods over the entire candidate document set, we introduce a recent TD based method to evaluate the feasibility of TD methods for IR.

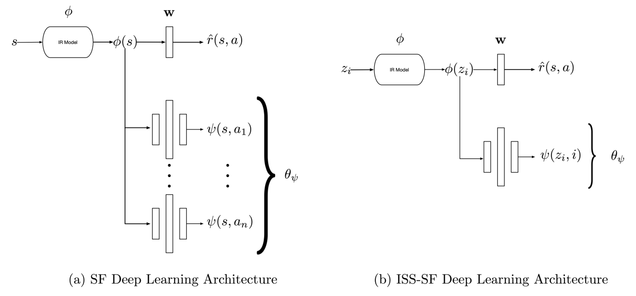

**Figure 3: Overview of the standard full state SF architecture (a) and the ISSTD adaptation for single Query-Document scoring (b).**

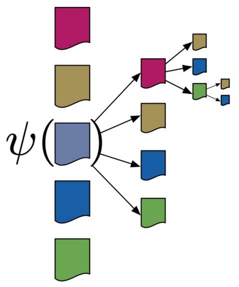

**Figure 4: The task of $\psi$ in the SF setting is to predict the future documents to be ranked given selecting the most relevant document at the current step. This forced representation learning coerces the SF based retrieval model to consider not only individual documents when determining relevance, but other documents it might see in a candidate ranked list.**

## 5 SUCCESSOR FEATURES FOR IR

With the machinery developed above, we are now able to incorporate successor feature (SF) learning [1], a powerful TD technique with unique properties which pair well with IR tasks. We introduce SF learning in the standard $s, a$ notation for clarity, and apply the spline interpolation discussed above during training. This framework offers two significant advantages over conventional Q-learning: (1) it formally separates representation and relevance judgements, reducing variance within the structure of the relevance portion, and (2) better captures the uncertainty of a current document's ranking due to the *successive* state representations [1, 10]. While the linear separation is not by itself novel, the idea of predicting the IR metrics of future documents to be ranked is a powerful tool for IR. This successor based representation compounds with the simplicity of linear relevance estimation by essentially forcing the model to not only predict the most relevant document at each time step but to accurately model the relevance of the rest of the expected documents to be ranked in the future, as shown in Figure 4. We provide an outline of this approach below.

The SF representation is based on the concept that the reward function $r(s,a)$, or document relevance, can be decomposed into an inner product of a state representation $\phi : \mathcal{S} \rightarrow \mathbb{R}^K$ and reward vector $\mathbf{w} \in \mathbb{R}^K$ such that

$$r(s,a) = \phi(s,a)^\intercal \mathbf{w}. \tag{17}$$

This representation is not restrictive to any environment as it can be trivially deconstructed to recover any reward function. We derive the successor representation, $\psi^\pi$ ,by incorporating $\phi$ and $\mathbf{w}$ into the standard $Q$-function to produce $\psi^\pi$.

$$Q^\pi(s,a) = \mathbb{E}[\Sigma_{i=0}^{\infty}\gamma^i r(s_i,a_i)|S_0 = s, A_0 = a, \pi] \tag{18}$$

$$= \mathbb{E}[\Sigma_{i=0}^{\infty}\gamma^i \phi(s_i,a_i)^\intercal \mathbf{w}|S_0 = s, A_0 = a, \pi] \tag{19}$$

$$= \mathbb{E}[\Sigma_{i=0}^{\infty}\gamma^i \phi(s_i,a_i)|S_0 = s, A_0 = a, \pi]^\intercal \mathbf{w} \tag{20}$$

$$= \psi^\pi(s,a)^\intercal \mathbf{w} \tag{21}$$

To conceptualize what $\psi$ means: in the tabular case such as Gridworld using one hot encodings, the $i^{th}$ component of $\psi^\pi$ is the discounted sum of occurrences of reaching $s_i$ of each possible transition while following $\pi$. For IR, this will be the discounted sum of document representations based on the ranking order determined by our retrieval model. As SF maintains linearity across time, any TD method can be used,

$$\psi^\pi(s,a) = \phi_(s,a) + \alpha[\gamma\psi^\pi(s',a) - \psi^\pi(s,a)], \tag{22}$$

and therefore can be trained in the same manner as Q-learning, referred to as SFQL. With Proposition 4.2 and Corollary 4.2.1, we see that this new approach can be applied within ISS framework (ISS-SFQL) for deep retrieval models. In the non-tabular case where a gradient is used to learn $\phi$, $\mathbf{w}$, and $\psi$, the optimization occurs via a two step process where $\psi$ is optimized via the loss function:

$$\mathcal{L}(\theta_\psi) = \mathbb{E}[||\phi(s) - \gamma \max_{a'} \psi(s',a') - \psi(\phi(s),a)||_2^2] \tag{23}$$

and $\mathbf{w}, \phi$ via:

$$\mathcal{L}(\theta_\mathbf{w}, \theta_\phi) = \mathbb{E}[||r(s,a) - \phi(s)^\intercal \mathbf{w}||_2^2]. \tag{24}$$

To clarify, while linear regression is being used in a portion of this state, it does not rely on any supervised labels and is used as a method to update $\psi$'s future trajectory estimates. Algorithm 1 provides an overview of the entire learning process and Figure 3 illustrates the overall framework. We adopt target networks and a memory buffer as recommended in Hessel et al. [7] to improve the stability of deep Q-learning agents during training.[1]

By defining a SF approach for ranking, we enforce search to be decomposed into a representation component and a reward component. Furthermore, $\psi^\pi(s,a)$ captures the expected future steps within its construction, forcing our model to predict a multi-step function even in sparse environments.

## 6 EXPERIMENTAL SETUP

To evaluate the efficacy of TD based updates for IR, we examine representative IR models of varying complexity as policies over MSMARCO passage, Yahoo L4, and InsuranceQA. These datasets were chosen due to the need to evaluate RL based methods on true deep neural architectures and provide an approximate upper bound of performance as true supervised labels are present.

[1]https://github.com/dscohen/ISSTD

**Algorithm 1:** Approach for ISS-SF.

```
Input: Memory replay $\mathcal{B}$, parameters $\theta_\phi, \theta_\mathbf{w}, \theta_\psi$,
  exploration probability $\epsilon \leq 1$, MDP: $(\mathcal{S}, \mathcal{A}, P, R, d_0, \gamma)$
for episode $l$ = 1 to $L$ do
    initialize $s \sim d_0$
    if Bernoulli($\epsilon$) then
        sample action $a$ uniformly
    else
        $\phi(s) = \{f_{IR}(z_i)\}_0^{|s|}$
        $a = \text{argmax}_i\ \psi(z_i, i)^\intercal \mathbf{w}$
    Store transition $(z, i, r(s,a), s')$ in $\mathcal{B}$
    Randomly sample minibatch from $\mathcal{B}$
    Update $\theta_\phi, \theta_\mathbf{w}$ via Eq. 24
    Update $\theta_\psi$ via Eq. 23
    $s \leftarrow s'$
```

### 6.1 Data

**MSMARCO:** [19] This collection is based on Bing queries and their corresponding results. Originally proposed for a question answering task, the annotated data was used to create a passage reranking task. The collection consists of 400M training tuples of query, relevant, and non-relevant passages with the development set containing ~6,900 queries with each query corresponding to the top 1,000 passages retrieved via BM25. A portion of the training set was partitioned to act as validation during training to fairly report results on the development set.

**Yahoo L4:** [27] Yahoo L4 consists of non-factoid questions which take the form of "Manner" questions. The collection consists of ~142,000 queries and corresponding answers. The train, dev, and eval splits were 80%, 10%, and 10% respectively.

**Insurance QA:** [3] This collection is a short text question-answer dataset from the insurance domain where questions are created from real user submissions, and the high quality answers come from insurance professionals. The dataset consists of 12,887 QA pairs for training, 1,000 pairs for validation, and two test sets containing 1,800 pairs on which we report the mean performance. For testing, each of the 1,800 QA pairs is evaluated with 499 randomly sampled candidate answers.

### 6.2 IR Policies – Network Architectures

To demonstrate the feasibility of a TD based RL approach for conventional information retrieval, we use three different representative architectures as policies which cover recurrent, convolutional, and transformer paradigms. We note the absence of a BERT based model as the experiments are to demonstrate the feasibility of IS-STD learning with varying architecture complexities. As BERT is used as a pretrained base [40], it does not offer additional insight. We specifically target deep neural architectures as they represent real-world models that require non-trivial optimization as opposed to feature-based learning to rank. In addition, said challenges will help highlight the differences in performance between REINFORCE, ISSQL, and ISSSF, and their approximate upper bound of supervised learning [28].

**MatchPyramid:** This model consists of an initial interaction matrix of the query and document's embeddings that acts as an input to a series of convolutional layers. While a deep network, this model has shown to perform well on only a small number of training queries, as demonstrated on Robust04 and other TREC collections [23].

**LSTM:** As these architectures have been used extensively for retrieval and are well studied [17, 22, 30], they provide a stable candidate to evaluate ISS-SFQL. The model consists of concatenating the query and document and feeding it into a bidirectional LSTM model. Max pooling over time is used prior to an upper feedforward network.

**Transformer Kernel:** Lastly, we introduce a state of the art model representative of the current neural retrieval architectures. This approach uses transformer modules with a kernel pooling approach to learn the relevance score between a query and candidate document and is the largest model of the architectures evaluated in this paper [8].

As $\phi$ is a multidimensional representation of a document, the standard architecture of the above models results in a scalar output. Therefore, we use the layers prior to the final output as $\phi$ and fix the output of each IR model to a fixed dimension. MatchPyramid and LSTM-Match use a 200 dimension final layer, and Transformer Kernel uses $\phi \in \mathbb{R}^{22}$. While significantly smaller than the other two approaches, this is due to the number of kernels used in the original paper and offers a salient and compressed representation of the query-document relation.

## 6.3 Baselines

As ISS-SFQL is an optimization method over a neural model, $\pi$, we evaluate performance against the current standard for RL in IR, REINFORCE [34, 37]. We also include pairwise hinge loss to demonstrate how well these neural architectures can perform under ideal training conditions, although this is not meant as a direct comparison.

## 6.4 Optimization

All models and optimization methods use the Adam optimizer with tuned learning rates via search over $[10^{-5}, 10^{-1}]$. We initialize all embeddings with GloVE [24] with a dimension of 300. Other hyperparameters for the models were taken from their corresponding works and demonstrated robustness across collections. The memory buffer for ISS-SFQL was constructed from $\{500, 2000, 10000, 20000\}$, and the target network and policy's updates per steps were selected from $\{1, 5, 10, 50\}$. The memory was not reset at each new episode. $\gamma$ was selected from $\{0.5, 0.7, 0.9, 0.99\}$. We discuss the sensitivity of the algorithm to these hyperparameters below.

## 7 RESULTS AND DISCUSSION

In this section, we investigate whether the results from Proposition 4.2 and Corollary 4.2.1 hold for real-world examples and can achieve competitive results with deep neural architectures. We furthermore evaluate the efficacy of SF for IR by examining cross-collection retrieval as a lifelong learning problem.

## 7.1 Performance Benchmarks

Examining the performance on Yahoo L4 and MSMARCO, we see consistent performance across models and training regimes as shown in Table 2. We remark on the competitive performance of ISS-SFQL, suggesting that ISSTD approaches are viable for IR tasks even in the case of deep neural models. While not the purpose of this paper, the surprising result of the close performance difference between the supervised pairwise hinge loss and ISS-SFQL suggests that RL methods can act as an effective training alternative to supervised approaches even in RL's most difficult environment – a sparse reward setting. Given the nature of instability of RL methods, the stability of supervised approaches, and the minimal hyperparameter tuning, this provides promising insight into situations where no supervised training signal is available, such as query reformulation [18] or online recommendation systems where the input is significantly large.

Of particular note is the poor performance of REINFORCE. We attribute the close to random performance of these policies due to the point collapse of the problem. While training, the distribution $\pi(s)$ would collapse to a single action despite our efforts due to the gradient variance introduced via Markov chain Monte Carlo sampling [28] and the nonlinear function approximation. We are careful to comment that this is not indicative that policy gradient methods are worse than TD approaches. However, by incorporating additional structures into a TD framework (SF), we can significantly increase the performance of RL based methods in cases where REINFORCE fails. While not examined in this work, ISS-Actor-Critic or ISS-DDPG could be alternative, more stable, options to use policy gradient in IR.

## 7.2 Convergence

Table 3 reflects the benefit of incorporating novel RL methods. ISS-SFQL is able to quarter the required number of episodes compared to ISS-Q-learning to converge to a stable policy with respect to performance on a validation set. In contrast, MDPRank's algorithm fails to converge at all when used for deep neural retrieval. Furthermore, ISS-SFQL only requires twice as many samples as the supervised approach while undergoing a bootstrap procedure.

## 7.3 Hyperparameter Sensitivity

Any RL algorithm has a substantial number of hyperparameters that can drastically impact performance. We highlight the key aspects in Figure 5. Following the discussion of the transformer's architecture being sensitive to noise, we observe a similar sensitivity to hyperparameters. The most poignant example of this is the update frequency of $\phi$, $\mathbf{w}$, and $\psi$. As the actual update to these values occurs via minibatch from the memory buffer, updating too frequently has the potential to introduce too much noise into these minibatches. By slowing down the update frequency such that the model changes once every $m$ steps, we increase the likelihood of sampling a more uniform batch to update.

This sensitivity is supported by the impact of replay size. We observe a decrease in performance concerning TK as the memory buffer size increases. As Liu and Zou [16] discuss the impact of buffers with respect to agent performance, it is not uncommon to see this sensitivity. We hypothesize that it is due to storing older

| Method | YahooL4 | | | MSMARCO | | | InsuranceQA | | |
|---|---|---|---|---|---|---|---|---|---|
| | LSTM | TK | MP | LSTM | TK | MP | LSTM | TK | MP |
| REINFORCE | 0.288 | 0.319 | 0.304 | 0.338 | 0.295 | 0.306 | 0.340 | 0.292 | 0.362 |
| ISS-Q-Learning | 0.409 | 0.421 | 0.416 | 0.331 | 0.325 | 0.374 | 0.419 | 0.435 | 0.426 |
| ISS-SFQL | **0.557**$^\dagger$ | **0.535**$^\dagger$ | **0.484**$^\dagger$ | **0.531**$^\dagger$ | **0.653**$^\dagger$ | **0.488**$^\dagger$ | **0.693**$^\dagger$ | **0.731**$^\dagger$ | **0.621**$^\dagger$ |
| Hinge loss | 0.550 | 0.586 | 0.487 | 0.551 | 0.687 | 0.481 | 0.697 | 0.722 | 0.620 |

**Table 2: Performance of training regimes over Yahoo L4, MSMARCO, and InsuranceQA data evaluated via MRR. $^\dagger$ denotes significance using Wilcoxon signed ranked test at $p < 0.05$.**

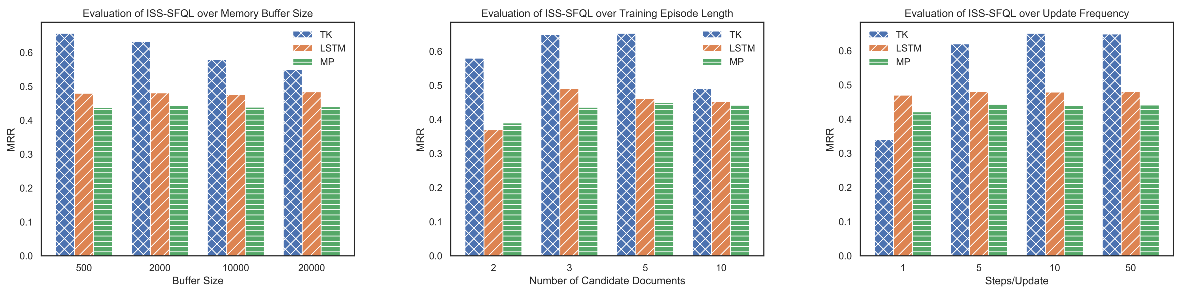

**Figure 5: Performance of ISS-SFQL across key hyperparameters on MSMARCO dataset.**

| Method | LSTM | TK | MP |
|---|---|---|---|
| REINFORCE | $\infty$ | $\infty$ | $\infty$ |
| ISS-Q-Learning | 210k | 410k | 205k |
| ISS-SFQL | 35k | 135k | 35k |
| Hinge loss | 20k | 90k | 15k |

**Table 3: Number of episodes (queries) needed to converge to a stable retrieval model on Yahoo L4. Evaluation was done every 5,000 episodes. $\infty$ denotes that the method diverges.**

transitions that no longer benefit the agent during the minibatch update. While the more stable LSTM and MP models are robust to this noise, the transformer-based architecture, TK, diverges.

Lastly, we examine the episode length, or initial size of $D_{ur}$ from the MDP formulation. For reference, when set to 2, this mirrors the bandit situation as the final document acts as the terminal state and no future decisions need to be considered. Therefore, as the agent is allowed to make more decisions and observe more documents, it can better determine what is a relevant document and what is not. This can be potentially attributed to the multi-step nature of $\psi$. Not only does it need to estimate the reward directly via $\mathbf{w}$, but the construction requires the prediction of the subsequent documents. Thus, when given a query and document $\phi(z_i)$, $\psi$ has to estimate what other similar documents are in the collection. The impact of this auxiliary task is shown in the episode length graph. The sharp decline of TK when the episode length equals 10 is due to the limited capacity of its $\phi$, which only has 22 dimensions by construction [8]. In this case, a slight modification of ISS-SFQL by expanding the number of kernels used might be a better alternative than directly truncating the final layer. A similar saturation occurs in the case of MP and LSTM models, although without the performance drop-off observed for TK.

## 8 CONCLUSION

We propose a novel, theoretically motivated framework to incorporate temporal difference based reinforcement learning for full text retrieval under a state partitioned function. We empirically demonstrate the flexibility of this approach by evaluating different neural retrieval architectures in a 'plug 'n play' format that does not require specific modifications or unique MDP formulations to handle the size of the state space.

## 9 ACKNOWLEDGEMENTS

We appreciate the valuable feedback provided by the anonymous reviewers. This work was supported in part by the Center for Intelligent Information Retrieval and in part by the Office of the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA) via AFRL contact #FA8650-17-C-9116 under subcontract #94671240 from the University of Southern California. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the ODNI, IARPA, or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect those of the sponsor.

## REFERENCES

[1] Andre Barreto, Diana Borsa, John Quan, Tom Schaul, David Silver, Matteo Hessel, Daniel Mankowitz, Augustin Zidek, and Remi Munos. 2018. Transfer in Deep Reinforcement Learning Using Successor Features and Generalised Policy Improvement. In *Proceedings of the 35th International Conference on Machine Learning (Proceedings of Machine Learning Research)*, Jennifer Dy and Andreas Krause (Eds.), Vol. 80. PMLR, Stockholmsmässan, Stockholm Sweden, 501–510. http://proceedings.mlr.press/v80/barreto18a.html

[2] Charles L.A. Clarke, Maheedhar Kolla, Gordon V. Cormack, Olga Vechtomova, Azin Ashkan, Stefan Büttcher, and Ian MacKinnon. 2008. Novelty and Diversity in Information Retrieval Evaluation. In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (Singapore, Singapore) *(SIGIR '08)*. Association for Computing Machinery, New York, NY, USA, 659–666. https://doi.org/10.1145/1390334.1390446

[3] Minwei Feng, Bing Xiang, Michael R. Glass, Lidan Wang, and Bowen Zhou. 2015. Applying Deep Learning to Answer Selection: A Study and An Open Task. *CoRR* abs/1508.01585 (2015). arXiv:1508.01585 http://arxiv.org/abs/1508.01585

[4] Ryosuke Furuta, Naoto Inoue, and Toshihiko Yamasaki. 2020. PixelRL: Fully Convolutional Network With Reinforcement Learning for Image Processing. *IEEE Trans. Multimedia* 22, 7 (2020), 1704–1719. https://doi.org/10.1109/TMM.2019.2960636

[5] Geoffrey J. Gordon. 1995. *Stable Function Approximation in Dynamic Programming.* Technical Report. USA.

[6] Geoffrey J. Gordon. 2000. Reinforcement Learning with Function Approximation Converges to a Region. In *Advances in Neural Information Processing Systems 13, Papers from Neural Information Processing Systems (NIPS) 2000, Denver, CO, USA*, Todd K. Leen, Thomas G. Dietterich, and Volker Tresp (Eds.). MIT Press, 1040–1046. http://papers.nips.cc/paper/1911-reinforcement-learning-with-function-approximation-converges-to-a-region

[7] Matteo Hessel, Joseph Modayil, Hado van Hasselt, Tom Schaul, Georg Ostrovski, Will Dabney, Dan Horgan, Bilal Piot, Mohammad Gheshlaghi Azar, and David Silver. 2018. Rainbow: Combining Improvements in Deep Reinforcement Learning. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, Sheila A. McIlraith and Kilian Q. Weinberger (Eds.). AAAI Press, 3215–3222. https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/17204

[8] Sebastian Hofstätter, Markus Zlabinger, and Allan Hanbury. 2020. Interpretable & Time-Budget-Constrained Contextualization for Re-Ranking. In *Proc. of ECAI*.

[9] Yujing Hu, Qing Da, Anxiang Zeng, Yang Yu, and Yinghui Xu. 2018. Reinforcement Learning to Rank in E-Commerce Search Engine: Formalization, Analysis, and Application. *arXiv:1803.00710 [cs]* (May 2018). http://arxiv.org/abs/1803.00710 arXiv: 1803.00710.

[10] David Janz, Jiri Hron, José Miguel Hernández-Lobato, Katja Hofmann, and Sebastian Tschiatschek. 2018. Successor Uncertainties: Exploration and Uncertainty in Temporal Difference Learning. *arXiv:1810.06530 [cs, stat]* (Oct. 2018). http://arxiv.org/abs/1810.06530 arXiv: 1810.06530.

[11] Vijay Konda and John Tsitsiklis. 2000. Actor-Critic Algorithms. In *SIAM Journal on Control and Optimization*. MIT Press, 1008–1014.

[12] Timothy P. Lillicrap, Jonathan J. Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. 2016. Continuous control with deep reinforcement learning. In *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, Yoshua Bengio and Yann LeCun (Eds.). http://arxiv.org/abs/1509.02971

[13] Aldo Lipani. 2016. Fairness in Information Retrieval. In *Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval* (Pisa, Italy) *(SIGIR '16)*. Association for Computing Machinery, New York, NY, USA, 1171. https://doi.org/10.1145/2911451.2911473

[14] Feng Liu, Huifeng Guo, Xutao Li, Ruiming Tang, Yunming Ye, and Xiuqiang He. 2020. End-to-End Deep Reinforcement Learning Based Recommendation with Supervised Embedding. In *Proceedings of the 13th International Conference on Web Search and Data Mining* (Houston, TX, USA) *(WSDM '20)*. Association for Computing Machinery, New York, NY, USA, 384–392. https://doi.org/10.1145/3336191.3371858

[15] Feng Liu, Ruiming Tang, Xutao Li, Weinan Zhang, Yunming Ye, Haokun Chen, Huifeng Guo, and Yuzhou Zhang. 2019. Deep Reinforcement Learning based Recommendation with Explicit User-Item Interactions Modeling. *arXiv:1810.12027 [cs]* (Oct. 2019). http://arxiv.org/abs/1810.12027 arXiv: 1810.12027.

[16] Ruishan Liu and James Zou. 2017. The Effects of Memory Replay in Reinforcement Learning. *arXiv:1710.06574 [cs, stat]* (Oct. 2017). http://arxiv.org/abs/1710.06574 arXiv: 1710.06574.

[17] John Miller and Moritz Hardt. 2019. Stable Recurrent Models. In *International Conference on Learning Representations*. https://openreview.net/forum?id=Hygxb2CqKm

[18] Ali Montazeralghaem, Hamed Zamani, and James Allan. 2020. A Reinforcement Learning Framework for Relevance Feedback. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval* (Virtual Event, China) *(SIGIR '20)*. Association for Computing Machinery, New York, NY, USA, 59–68. https://doi.org/10.1145/3397271.3401099

[19] Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. 2016. MS MARCO: A Human Generated MAchine Reading COmprehension Dataset. *CoRR* abs/1611.09268 (2016). arXiv:1611.09268

[20] Rodrigo Nogueira and Kyunghyun Cho. 2017. Task-Oriented Query Reformulation with Reinforcement Learning. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*, Martha Palmer, Rebecca Hwa, and Sebastian Riedel (Eds.). Association for Computational Linguistics, 574–583. https://doi.org/10.18653/v1/d17-1061

[21] Anupiya Nugaliyadde, Kok Wong, Ferdous Sohel, and Hong Xie. 2017. Reinforced Memory Network for Question Answering. 482–490. https://doi.org/10.1007/978-3-319-70096-0_50

[22] Hamid Palangi, Li Deng, Yelong Shen, Jianfeng Gao, Xiaodong He, Jianshu Chen, Xinying Song, and Rabab K. Ward. 2015. Deep Sentence Embedding Using the Long Short Term Memory Network: Analysis and Application to Information Retrieval. *CoRR* abs/1502.06922 (2015). http://arxiv.org/abs/1502.06922

[23] Liang Pang, Yanyan Lan, Jiafeng Guo, Jun Xu, and Xueqi Cheng. 2016. A Study of MatchPyramid Models on Ad-hoc Retrieval. *CoRR* abs/1606.04648 (2016). arXiv:1606.04648 http://arxiv.org/abs/1606.04648

[24] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. GloVe: Global Vectors for Word Representation. In *EMNLP*. 1532–1543. http://www.aclweb.org/anthology/D14-1162

[25] Stephen Robertson and Hugo Zaragoza. 2009. The Probabilistic Relevance Framework: BM25 and Beyond. *Found. Trends Inf. Retr.* 3, 4 (April 2009), 333–389. https://doi.org/10.1561/1500000019

[26] Ashudeep Singh and Thorsten Joachims. 2019. Policy Learning for Fairness in Ranking. *CoRR* abs/1902.04056 (2019). arXiv:1902.04056 http://arxiv.org/abs/1902.04056

[27] Mihai Surdeanu, Massimiliano Ciaramita, and Hugo Zaragoza. 2008. Learning to Rank Answers on Large Online QA Collections. *Proceedings of the ACL-08: HLT* June (2008), 719–727.

[28] Richard S. Sutton and Andrew G. Barto. 2018. *Reinforcement Learning: An Introduction.* A Bradford Book, Cambridge, MA, USA.

[29] Csaba Szepesvári and William D. Smart. 2004. Interpolation-based Q-learning. In *ICML '04*. ACM Press, Banff, Alberta, Canada, 100. https://doi.org/10.1145/1015330.1015445

[30] Di Wang and Eric Nyberg. [n.d.]. A Recurrent Neural Network based Answer Ranking Model for Web Question Answering. In *WebQA Workshop, SIGIR '15, Santiago, Chile.*

[31] Jun Wang, Lantao Yu, Weinan Zhang, Yu Gong, Yinghui Xu, Benyou Wang, Peng Zhang, and Dell Zhang. 2017. IRGAN: A Minimax Game for Unifying Generative and Discriminative Information Retrieval Models. *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval - SIGIR '17* (2017), 515–524. https://doi.org/10.1145/3077136.3080786 arXiv: 1705.10513.

[32] Yu Wang and Hongxia Jin. 2019. A Deep Reinforcement Learning Based Multi-Step Coarse to Fine Question Answering (MSCQA) System. *Proceedings of the AAAI Conference on Artificial Intelligence* 33, 01 (July 2019), 7224–7232. https://doi.org/10.1609/aaai.v33i01.33017224 Number: 01.

[33] Christopher J. C. H. Watkins and Peter Dayan. 1992. Q-learning. In *Machine Learning*. 279–292.

[34] Zeng Wei, Jun Xu, Yanyan Lan, Jiafeng Guo, and Xueqi Cheng. 2017. Reinforcement Learning to Rank with Markov Decision Process. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, Shinjuku Tokyo Japan, 945–948. https://doi.org/10.1145/3077136.3080685

[35] Ronald J. Williams. 1992. Simple Statistical Gradient-Following Algorithms for Connectionist Reinforcement Learning. *Mach. Learn.* 8, 3-4 (May 1992), 229–256.

[36] Lijun Wu, Fei Tian, Tao Qin, Jianhuang Lai, and Tie-Yan Liu. 2018. A Study of Reinforcement Learning for Neural Machine Translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Brussels, Belgium, 3612–3621. https://doi.org/10.18653/v1/D18-1397

[37] Long Xia, Jun Xu, Yanyan Lan, Jiafeng Guo, Wei Zeng, and Xueqi Cheng. 2017. Adapting Markov Decision Process for Search Result Diversification. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval - SIGIR '17*. Shinjuku, Tokyo, Japan, 535–544. https://doi.org/10.1145/3077136.3080775

[38] X. Xing and D. E. Chang. 2019. Deep Reinforcement Learning Based Robot Arm Manipulation with Efficient Training Data through Simulation. In *2019 19th International Conference on Control, Automation and Systems (ICCAS)*. 112–116.

[39] Jun Xu, Zeng Wei, Long Xia, Yanyan Lan, Dawei Yin, Xueqi Cheng, and Ji-Rong Wen. 2020. Reinforcement Learning to Rank with Pairwise Policy Gradient.

In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, SIGIR 2020, Virtual Event, China, July 25-30, 2020*, Jimmy Huang, Yi Chang, Xueqi Cheng, Jaap Kamps, Vanessa Murdock, Ji-Rong Wen, and Yiqun Liu (Eds.). ACM, 509–518. https://doi.org/10.1145/3397271.3401148

[40] Zeynep Akkalyoncu Yilmaz, Shengjin Wang, Wei Yang, Haotian Zhang, and Jimmy Lin. 2019. Applying BERT to Document Retrieval with Birch. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019 - System Demonstrations*, Sebastian Padó and Ruihong Huang (Eds.). Association for Computational Linguistics, 19–24. https://doi.org/10.18653/v1/D19-3004

[41] Wei Zeng, Jun Xu, Yanyan Lan, Jiafeng Guo, and Xueqi Cheng. 2018. Multi Page Search with Reinforcement Learning to Rank. In *Proceedings of the 2018 ACM SIGIR International Conference on Theory of Information Retrieval (ICTIR '18)*. Association for Computing Machinery, Tianjin, China, 175–178. https://doi.org/10.1145/3234944.3234977

[42] Xiangyu Zhao, Long Xia, Jiliang Tang, and Dawei Yin. 2019. Reinforcement Learning for Online Information Seeking. *arXiv:1812.07127 [cs]* (April 2019). http://arxiv.org/abs/1812.07127 arXiv: 1812.07127.