A Study of Temporal Citation Count Prediction using Reinforcement Learning

Hengshuai Yao, Davood Rafiei, and Rich Sutton

***Abstract*—In a recently studied problem, Yan et. al. (2011) studied predicting the count of citations that already happened using pair-wise machine learning. Predicting the number of future citations based on only the past data is obviously more practical and more challenging. In this paper, we study the problem of temporal prediction of citation counts for academic papers. We propose a model-free method and a model-based method, respectively for predicting citation counts in both long and short terms. We extend the citation count measure to a general value function, which forms the basics for our application of reinforcement learning (RL). Our methods are based on two RL algorithms, *least-squares temporal difference* (LSTD) and *linear Dyna*. Our methods use quite a few novel features including those from citing papers and historical citation counts as well as those based on authors, keywords, and venues. Empirical results show that temporal prediction has its unique difficulties, and the pair-wise supervised learning methods can be unstable. Both our methods produce stable and accurate predictions. In addition, results also suggest that, unlike previous citation count prediction results, temporal prediction of citation count in a longer time span is less accurate. This highlights the need to study the temporal prediction problem and develop accurate predictors.**

***Index Terms*—multi-step time series prediction, citation count, reinforcement learning, temporal difference learning methods**

I. Introduction

Modeling and predicting time series is an important problem that has a wide range of applications. For example, in weather forecasting one estimates several variables such as the probability of raining, the probability of hails, the direction of wind, etc. An important feature of real-world prediction problems is that one has access to a sequence of certain observations and predictions are made strictly based on the past. In the weather forecasting problem one can observe, for example, the current local temperature, the weather yesterday, etc. A predictor is hence expected to take advantage of observations in making accurate predictions about the future.

In this paper, we study the problem of predicting citation counts in large academic networks. The citation count measures the importance of a paper by the number of citations it receives during a period [8]. A well practiced extension is the *impact factor* used by Thomson Reuters Journal Citation Reports, which measures a journal by the average number of citations received by the articles published. Though new methods of ranking papers and scholars keep emerging, citation count is still the most popular measure for academic evaluation, partly because it is intuitive, simple, yet effective.

Hengshuai Yao, Davood Rafiei, and Rich Sutton are with the Department of Computer Science, University of Alberta, Edmonton, AB, Canada, T6G2E8. E-mail: hengshua,drafiei,sutton@cs.ualaberta.ca.

Moreover, modeling trends of citation count is important also because it provides insights into the quality and trends of scientific research. The 2003 KDD Cup [9] contains a task of predicting the numbers of citations for a group of physics papers. The challenge of modeling citation counts is that they are highly dynamic and grow at different rates for individual papers. Indeed, the general problem of predicting trends of dynamic data is a challenging topic in data mining, and is driven by practical applications. For example, Netflix put a lot of emphasis on predicting user interests in movies and used to have a competition with one-million-dollar reward to increase their accuracy by 10%. In guaranteed display advertising it is critical to accurately estimate future user visits for the benefits of both advertisers and publishers [1]. In search engines, predicting query count helps build language models to help detect trends early [10].

In general, in this paper we propose a multi-step formulation of predicting a temporal function, as shown in Figure 1. A *state*, which can be defined arbitrarily according to applications, has some associated value that depends on the future. Given a history of temporal observations of a state, we would like to predict its value starting from any time. We also would like to predict for states whose histories are unavailable, by learning from their similar states—in the machine learning language—we would like to *generalize*. In this paper, we focus in particular on predicting future citation counts for research papers in academic networks. In this case, a state is a paper, the value to predict is the number of future citations that the paper receives since a given year. We observe the features and the numbers of citations of a group of papers in a number of years, and the goal is to learn a predictor for future citation count. We deal with predicting both short and long term citation counts. It is important to note that our problem is very different from the one studied by Yan et.al. [22]. Their problem is to predict the citation count that already happened based on historical data. For example, in their one-year prediction experiment, a data set of 10,000 papers were taken as the training set, and another set of 10,000 papers taken as the test set, both collected from year 2009. Intuitively, their problem is somehow a *spatial prediction* problem which generalizes only in papers. Their problem is well motivated by machine learning applications in which one wants to learn a generalization function from training data. In our problem, predictions about the future are strictly based on information from the past. Thus our problem is not only a spatial prediction problem but also a *temporal prediction* problem, which generalizes in both papers and time and is obviously more practical and challenging. The temporal aspect has unique difficulties as we will show in this paper.

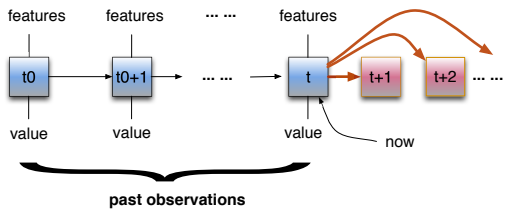

Fig. 1: The general problem we study. We are to predict the future value of a "state" starting from a given year in a certain (potentially variable) number of years. We have a sequence of past observations for the state. Each observation consists in a temporal value ("reward") and some associated features at each time step.

Our Problem. Let t ($t \geq 0$) be our current time step. Given a historical observation sequence about a variable x, $\{x_j\}_{0 \leq j \leq t-1}$, where x_j is a vector of observations at time step j, the task is to predict a scalar value dependent on x and time step t. We will refer to this problem as a *temporal prediction* problem given its time-based nature.

Example 1. A robot has many sensors. Each sensor enables the robot to observe one particular aspect of the environment. For example, a sensor detects the brightness, another monitors the direction and the speed of the robot, while another measures the distance to the nearest obstacle in front of it; etc. One interesting prediction problem is how likely the robot hits an obstacle in the next few seconds based on the sensor signals, e.g., see [13].

Example 2. Regarding the specific application in this paper, we can observe many features of an article at a given year, the task is to predict the number of citations that the paper receives in k years. Note here $k > 0$, which potentially goes to infinity. If k is small, we call it a *short-term* prediction; otherwise a *long-term* prediction.

The contribution of this paper are two principled methods for solving the problems, which are new to the literatures of data mining and information retrieval. We explore the use of reinforcement learning (RL), and propose a model-free and a model-based method to this goal. The model-free method extends from the so-called *least-squares temporal difference* (LSTD) learning due to Bradtke and Barto [4], and Boyan [3]. We use LSTD to predict the long-term future citation counts. LSTD builds two structures on successive observations, and performs a psudo-least-squares procedure based on the structures. The model-based method is a generalized form of *linear Dyna*, which is a more recent RL architecture due to Sutton and his colleagues [20]. Our extension can predict future citation counts in both short and long terms. The key of the method is a transition kernel among features of papers at different times. This transition kernel models how citation counts and paper features change year by year. With the kernel, we can describe multi-step trends of citation counts through a projection procedure. We use quite a few novel features for the problems, including those from citing papers and historical citation counts as well as those based on authors, keywords, and venues. Experimental studies were performed on a large computer science database (DBLP) with more than one million papers.

The remainder of this paper is organized as follows. Section II provides a necessary and minimal background on relevant RL concepts. In Section III, we formulate the citation count measure as a simple value function in RL, which provides the basics for this paper. In Section IV, we formally define the studied prediction problems, and extend LSTD and linear Dyna to solve them. Section VI-A describes the features we use for the problems. We discuss related work in Section V. Section VII concludes the paper.

II. Background

In this section, we introduce Markov reward processes (MRPs), linear value function approximation, LSTD, and linear Dyna architecture for MRPs.

A. MRPs

We consider a discrete-time, finite state Markov chain. The state space is $\mathbb{S} = \{1, 2, \ldots, N\}$. The transition probability from a state s to a state s' is $\mathcal{P}(s, s')$, for $\forall s, s' \in \mathbb{S}$. At a time step t, a scalar reward r_t is given. The value of a state s is defined as the expected long-term rewards starting from the state itself:

$$V(s) = \mathbb{E}\left\{\sum_{t=0}^{\infty} \gamma^t r_t\right\}, \quad s_0 = s,$$

where $\gamma \in [0, 1)$ is a discount factor, and $\mathbb{E}$ is the expectation operator taken with respect to the distribution of the states.

B. Linear Function Approximation

RL is advantageous in the ability of learning the value function using function approximation, which provides generalization among states and compact representation. *Linear function approximation* refers to approximating V with a linear parametric function:

$$\hat{V}(i) = \phi(i)^\top \theta,$$

where θ is a vector of parameters, and $\phi(i)$ is a vector of features of state i, for $\forall i \in \mathbb{S}$. The feature vectors are often constructed from a number of feature functions. Let $\mathbb{R}$ be the real number space. Given d ($d \leq N$) feature functions $\varphi_j(\cdot) : \mathbb{S} \mapsto \mathbb{R}$, $j = 1, \ldots, d$, the *feature vector* of state i is $\phi(i) = [\varphi_1(i), \varphi_2(i), \ldots, \varphi_d(i)]^\top$. Note that though the function is linear in the parameters, a feature mapping can be any nonlinear function in general.

C. LSTD

The algorithm is shown in Algorithm 1 [4, 3]. The algorithm accumulates coefficients $A \in \mathbb{R}^{d \times d}$ and $b \in \mathbb{R}^d$ across episodes, and solves the linear system to obtain an approximation of the value function, $\hat{V} = \Phi\theta$. The vector z is called the *eligibility trace*, wherein $\lambda \in [0, 1]$ is called the *eligibility trace factor*. [1] The eligibility trace weights historical

[1] Note here we used the most common eligibility trace [17]. There are also other eligibility traces, e.g., see [16].

Input: The observation sequence D_t.
Output: An approximate value function $\hat{V} = \Phi\theta$.

Set $t = 0$, Initialize A and b
for *each episode* **do**
  Select an initial state s_0
  $z_0 = \phi_0$ /* $\phi_t = \phi(s_t)$ */
  for *each time step* t **do**
    Observing s_{t+1} and r_t
    $A = A + z_t(\gamma\phi_{t+1} - \phi_t)^\top$
    $b = b + z_t r_t$
    $z_{t+1} = \lambda\gamma z_t + \phi_{t+1}$
  end
end
Solve $A\theta + b = 0$

Algorithm 1: LSTD(λ) for MRPs.

features exponentially, with more decaying on more distant states, controlled by λ. If $\lambda = 0$, the corresponding LSTD structures are the one-step correlations between successive features, and features with the immediate rewards. If $\lambda = 1$, the corresponding LSTD algorithm is shown to be equivalent to a linear regression procedure [3]. Intermediate λs smooth historical observations. It can be shown that LSTD(λ) minimizes a projected Bellman error function that is dependent on λ. In practice there are often cases where an intermediate value performs best [18]. In a summary, the input of LSTD is a sequence of snippets:

$$D_t = \{< \phi(s_j), \phi(s_{j+1}), r_j > | j = 0, 1, \ldots, t-1\},$$

where $\phi(s_j)$ and a scalar reward r_j are our observations at time step j. The output of LSTD is an approximate value function.

D. Linear Dyna

Dyna is an integrated architecture for estimating a value function, in which online learning and planning proceed simultaneously. It is based on a lookup table over the states, and has no generalization capability [19]. *Linear Dyna* extends Dyna with a compact linear world model and comes up with an approximation to the value function [20]. To understand linear Dyna, it helps understand the so-called *linear model* [20] first. In the case of MRPs, the *linear model* is defined by a pair $\langle F, f \rangle$, where $F \in \mathbb{R}^{d\times d}$ is a $d \times d$ matrix and $f \in \mathbb{R}^d$ is a d-dimensional vector. For any given state $s \in \mathbb{S}$, $F\phi(s)$ estimates the expected feature vector of the state $s' \sim \mathcal{P}(s, \cdot)$, while $f^\top\phi(s)$ estimates the expected reward of leaving state s in one step. For a good linear model we expect that with $s' \sim \mathcal{P}(s, \cdot)$,

$$F\phi(s) \approx E[\phi(s')] \quad \text{and} \quad f^\top\phi(s) \approx E[\mathcal{R}(s, s')],$$

where the expectation is taken with respect to the distribution of s' given s. In the remainder of this paper, we call F the *linear transient model* and f the *linear reward model*.

Linear Dyna integrates a model free algorithm called Temporal Difference (TD) learning, a modeling procedure that estimates the linear model, and a planning procedure that solves the linear model in a few loops. The complete linear Dyna algorithm for MRPs is shown in Algorithm 2. The

Set $t = 0$; Initialize θ_0, F_0 and f_0
Select an initial state s_0
for *each time step* t **do**
  Observe s_{t+1} and r_t
  TD(0): /* α_t is a step-size, and ϕ_t is short for $\phi(s_t)$ */
    $\theta_{t+1} = \theta_t + \alpha_t(r_t + \gamma\phi_{t+1}^\top\theta_t - \phi_t^\top\theta_t)\phi_t$
  Update the linear model $\langle F_t, f_t \rangle$
  Set $\tilde{\theta}_0 = \theta_{t+1}$
  repeat for $p = 0, 1, \ldots, \tau$ /*Planning*/
    Sample a feature vector $\tilde{\phi}_p$
    $\tilde{\phi}_{p+1} = F_{t+1}\tilde{\phi}_p$
    $\tilde{r}_p = \tilde{\phi}_p^\top f_{t+1}$
    $\tilde{\theta}_{p+1} = \tilde{\theta}_p + \alpha_p(\tilde{r}_p + \gamma\tilde{\theta}_p^\top\tilde{\phi}_{p+1} - \tilde{\theta}_p^\top\tilde{\phi}_p)\tilde{\phi}_p$
  end
  Set $\theta_{t+1} = \tilde{\theta}_{\tau+1}$
end

Algorithm 2: Linear Dyna for MRPs. The input and the output are the same as LSTD.

model-free part is irrelevant to the present paper. The key ideas of linear Dyna are summarized below. Firstly, the linear model predicts the expected next feature vector and the expected reward given a feature vector. Thus the linear model contains a one-step dynamics of transitioning in the feature space. In this sense, it helps to think of the linear model as a standard supervised learning model. Put it in the context of the citation count prediction problem, for example, applying the linear transient model to the feature vector of a paper in 2000, we get a prediction for the feature vector of the paper in the next year. Secondly, by iterating the linear model multiple times one can predict multiple steps. The linear model was shown to help speed up learning and control [20]. It is also shown that solving the linear model gives the same approximate value function as the model-free TD learning in the limit [14]. However, it may be rarely noticed that the linear model and projecting with it to do multi-step predictions generalizes beyond reinforcement learning.

III. The Citation-to-go Functions

The key observation leads to this article is a simple and natural extension of the citation count measure to a special value function in RL. In particular, we define the "value" of a paper s at time t by the sum of discounted numbers of citations in all the subsequent years:

$$V(s, t) = \sum_{q=t}^{\infty} \gamma^{q-t} c_q, \quad \gamma \in [0, 1)$$

where c_q is the number of citations the paper receives in a year q. That is, the reward signal here is the yearly citation count for this problem. Note that when t is the publication year of the paper and γ approaches one, $V(s, t)$ will be virtually close to the total number of citation counts for the paper. Introducing discounting gives a time weighting. When ranking papers according to V with discounting, it favors papers with many citations that occur immediately after they are published. This may lead to the discovery of relatively newly published papers easier for readers— an important problem people have been concerned about in recent years, e.g., see [12]. However, we do

not explore this hypothesis in this paper, but instead use a large discount factor (and smaller than one) to have a reasonably close approximation to the citation count measure. We call V the *citation-to-go (CTG)* function for short, reminiscent of the name *cost-to-go* function widely used in operation research [2]. The CTG satisfies a simple relationship which is called *Bellman equation* in the literatures of dynamic programming and RL. In particular, for a paper s,

$$V(s,t) = \sum_{q=t}^{\infty} \gamma^{q-t} c_q = c(s,t) + \gamma V(s,t+1),$$

where $c(s,t) = c_t$. That is, the citation-to-go of a paper from a specific year on is equal to the number of citations received by the paper in the year plus the discounted citation-to-go from the next year. Bellman equation is exploited by various RL algorithms. In this paper, our algorithms will also take advantage of this fact.

In practice, we never have data up to the infinite future. A practical extension is to truncate CTG:

$$V_T(s,t) = \sum_{q=t}^{T} \gamma^{q-t} c_q,$$

where T is an integer such that $t \leq T < \infty$. Clearly, $V_T(s,t)$ is the sum of discounted $T-t+1$ years of citation counts since year t for paper s. With a discount factor smaller than one, when T is large enough, V_T is a reasonable approximation to V since the tails are small. V_T also satisfies a Bellman equation. When $\gamma = 1$, the tails will be significant. In this case, not only V isn't well defined but also we aren't able to study V via V_T. This is why we do not use $\gamma = 1$. In short, our tasks are to predict the original discounted CTG function and the truncated discounted CTG function.

IV. Modeling Citation-to-go

Before proceeding, let us remind that by defining the CTG functions, we have effectively given a Markov formulation for the citation count prediction problems. At each time step (in this case a year), we observe the current state of a paper. Time moves on to the next year, and we observe the most recent state of the paper as well as the citations it receives during this one-year period. The next year's state depends only on the current state of the paper, i.e., the problem is *Markovian*. Notably, there are multiple episodes that proceed simultaneously (one episode for an individual paper). Adopting the notations from Section II, we predict the CTG functions with a set of features $\{\varphi_j\}_{j=1,2,\ldots,d}$. The state of a paper s at year t is mapped into a feature vector $\phi(s_t)$.

A. The Prediction Problems

An observation history for a paper s from year t_0 to year t is,

$$O(s,t_0,t) = \{\langle \phi(s_j), c_j, \phi(s_{j+1}) \rangle, j = t_0, t_0+1, \ldots, t\}.$$

Note that a paper's state changes from time to time, and s_{t_0}, s_{t_0+1}, ...are the status of the same paper s during the period. The data we have for both tasks are the past observations of some sampled papers, which is, $\mathbb{D} = \{O(s,t_0,t), | \text{ for some } s \in \mathbb{S}\}$. Note that this falls into the general problem that we described in Section I. In the long-term prediction task, the goal is to predict the CTG. We tackle this task by the method of LSTD. In the short-term prediction task, the goal is to predict the truncated CTG in a variable number of years. We solve this task by an extension of linear Dyna. Note that for both tasks, we mean that predictions are made strictly for the future, i.e., future citation counts starting from year t (where t is the last year for which we have data).

B. A Supervised Learning Approach

For the observation history $O(s,t_0,t)$ of paper s, a natural supervised learning approach forms a pair $\langle \phi(s_{t_0}), R^t_{t_0} \rangle$, where $R^t_{t_0}$ is the sum of the discounted future citations from year t_0 to year t. Then supervised learning methods are applied to learn a regression function. This is a natural extension of the approach taken in [6], [11], and [22] to the temporal prediction problem. This will be referred to as the *singly paired* (or *pair-wise* for short) approach.

C. LSTD(λ)

However, the above supervised learning predictor ignores the intermediate feature vectors and citations. Here is another formulation that considers the intermediate changes:

$$\phi(s_{t_0}) \to R^t_{t_0}; \quad \phi(s_{t_0+1}) \to R^t_{t_1}; \quad \ldots; \phi(s_t) \to R^t_t,$$

in which the left side is treated as the input and the right side is the output. One can then apply any supervised learning method to these pairs for all training papers. Interestingly, in the case of linear regression, this can be shown to be equivalent to LSTD(1) [3]. We used LSTD(λ) for predicting the long-term citation counts. A practical issue of implementing LSTD(λ) is that the matrix may be ill conditioned given finite samples. Thus some regularization is often necessary. A widely used practice is to initialize the matrix to $-\delta I$, where δ is a positive scalar and I is the identity matrix, since the matrix is negative definite in the limit. This can be shown equivalent to using L-2 regularization for LSTD.

D. A Projection Approach

LSTD(λ) does not predict short term, at least it was not designed for that purpose. To make short-term predictions, we propose a multi-step projection procedure based on linear Dyna, comprising the following two major steps.

Learning. With a state transition from s_t to s_{t+1}, we update our estimate for a linear model $(F_{d\times d}, f_{d\times 1})$. Basically F models the transitions among the feature vectors of papers, and f models the one-year citation count dynamics. Given a feature vector of some paper at a year, applying F predicts the expected feature vector of the paper in the next year, and applying f predicts the expected number of citations in the next year. At a year t, observing the new feature vector, $\phi(s_{t+1})$, and the number of citations within the year, $c(s,t)$, we update the linear model by

$$\Delta F = \beta \left[\phi(s_{t+1}) - F\phi(s_t)\right] \phi(s_t)^\top,$$

For each paper s to predict from year t to year $t+T$
  Initialize $\hat{V}(s,t,T)=0$
  Set $\hat{\phi}_t = \phi(s_t)$
  For $k = t, t+1, \ldots T$
    $\hat{c}_{k+1} = f^\top \hat{\phi}_k$
    $\hat{\phi}_{k+1} = F\hat{\phi}_k$
    $\hat{V}(s,t,T) = \hat{V}(s,t,T) + \hat{c}_{k+1}$
  end
end

Algorithm 3: Dyna-based projection method for predicting the citation-to-go with the linear model.

and

$$\Delta f = \beta \left[c(s,t) - f^\top \phi(s_t)\right] \phi(s_t),$$

where β is a step-size.

The f is nothing but a simple predictor about the one-year citation count. The linear transient model F is more interesting because there are multiple predictions that can be made according to it. Figure 2 illustrates the idea. In fact, each row of F is a predictor. In the example shown in the figure, the first row of F is a predictor for the number of citations in one year for the first author, the second row predicts the number of citations in one year for the last author given a feature vector; etc. Thus F is like a single-layer perceptron. This insight is new to the RL literature as well.

Projection. To model multi-step growth of citation counts, one continuously projects the feature vectors with F along the way, and combines with f to form the predictions for the citation count year by year. For example, suppose we are to make a 2-year prediction for the citation count of a paper s. We are provided with the features of the paper at the year, $\phi(s_t)$, which is the starting point for multi-step projections. The next year's citation count is predicted as, $\hat{c}_1 = f^\top \phi(s_t)$. To make predictions about the citation count of the second year (i.e., year $t+2$), we have to know the features of the paper at year $t+1$. However, $\phi(s_{t+1})$ is not available because it is about the future. This is where F plays a role. We do not have $\phi(s_{t+1})$, but we can predict it with F, in particular, by the projection, $\hat{\phi}_{t+1} \stackrel{def}{=} F\phi(s_t)$. Then the citation count of the second year is naturally predicted as $\hat{c}_2 = f^\top \hat{\phi}_{t+1}$. That is, we use a prediction (i.e., that of the next feature) to generate another prediction (i.e., that of the number of citations for the year after the next). More steps of predictions are similar. Algorithm 3 shows the details of this procedure. It helps to think of the algorithm as using the linear model to "grow" the citation count up to a future year by extrapolating for a certain number of steps from a feature vector.

To predict for different numbers of years, this approach requires only one-time training (which involves learning the linear model), while the pair-wise supervised approach has to train a separate predictor for different numbers of years.

V. Related Work

There are a few recent papers dealing with citation count predictions. As mentioned earlier, Yan et. al. studied the problem of generalizing predictions about the citation count

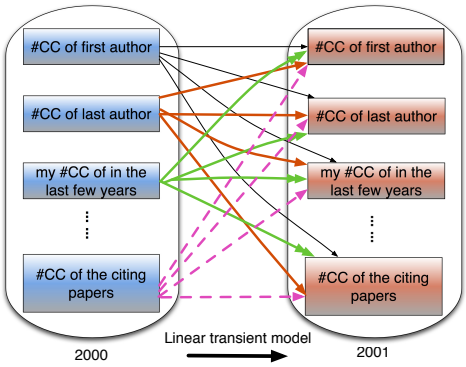

Fig. 2: Illustration of the linear transient model. The linear transient model can be viewed as a collection of multiple predictors. For example, the left side can be the feature vector of some paper in year 2000. The right side is then the prediction of its feature vector in the next year according to the model.

in the same period [22]. Fu et. al.'s is the earliest relevant work we found [7, 6]. They built prediction models that predict citation counts of biomedical publications within a horizon of ten years. Their models employed content and bibliometric based features in combination with support vector machine methods. In [11] they built predictors for articles in the *Bioinformatics* journal within four years of publication, using features from tokens in the abstracts and journal sections. Unfortunately, the problem studied in all these work is to predict the number of citations that already happened. While their problem is well motivated by machine learning and other literatures, predicting about future citation counts is certainly more interesting.

There are also some citation count prediction methods that are not obviously machine learning based. These methods usually involve models constructed from crafted heuristics [5, 15]. However, these models are hand tuned, thus leaving how to set their parameters a pressing question.

VI. Empirical Results

We conducted experiments on DBLP, a large computer science citation graph. The DBLP collection we use involves about one million authors, seven thousand venues, one and a half million papers and two million cross citations [21].

A. Features

We spent quite a bit effort in engineering the features. In particular, the features of a paper at a specific year are as follows.

1) The number of citations for the venue where the paper is published. Similar to the venue rank features [22], this encodes the reputation of venues.
2) The number of papers that are published at the venue until the year.
3) The impact factor of a venue, which is defined by the number of citations divided by the number of papers published at the venue until the considered year.

4) Venue life, which is the number of years that the venue has been around until the year. We estimated the establishment year of a venue by the earliest year when there are one or more papers published at it according to the database. This estimation works very well in our experience.
5) The author features of a paper comprise the maximum, average, and sum in the author properties of the papers. An author's properties are the citation count, impact factor, and the number of coauthors of the author. This feature provides a generalization over authors.
6) Identity features. Feature 5 is compressed, from which the identity of the authors is not recoverable. It is interesting to see whether identity-based features perform better. Each author occupies an entry of the feature vector, the feature value corresponding to the citation count of the author until the considered year.
7) Content features from the titles were also combined. In particular, about $300,000$ keywords were first extracted using NLTK (http://www.nltk.org). They were measured in a similar way to author features, corresponding to the number of containing papers, citation count and impact factor, and number of concurrent keywords, respectively. They were encoded also in a similar way to the author features.
8) Citation features. The citation features are based on the intuition that if a paper is cited by a famous author or a popular paper then it may be likely to receive many future citations. In particular, we encoded a paper at a specific year with the latest numbers of citations of the papers that cite it, together with the latest numbers of citations of the authors of these citing papers.
9) Historical citation features. Up to 10 years of historical citation count data for each paper were considered if they are available.

In addition, a constant feature was also used. Encoding papers with author, venue, and content information is a widely known practice in machine learning based classification. For citation count prediction, many of them have been practiced by [22] and [6]. Feature 2 to Feature 5 are mildly new for the problems. Feature 7 to Feature 9 are brand new, some of which are very helpful to the problems as will be seen in the experiment section.

B. The Long-term Prediction

Experiment setup. Throughout this experiment we are in the year 2002 ("now"). This is to ensure that there will be sufficient data to evaluate our predictor about the future. Training papers were those published before 1990. The training period considered is between year 1990 and "now". The discount factor is 0.9. After applying LSTD to the training set, we would like predict the sum of discounted numbers of future citations up to 2011. Specifically, this means with the learned weights and the feature vector of some paper in 2002, we should be able to make a prediction of the citation-to-go until 2011 for the paper. The true citation counts for the remaining years (2002-2011) were used in computing the (truncated) CTG function for evaluation. We varied the test set, and studied the cases of predicting for both old papers and newly published papers.

We compared LSTD against the pair-wise supervised learning approach. The input are the feature vectors of the training papers at year 1990, and the output are the truncated CTGs of the papers between 1990 and 2002. In particular, linear regression and support vector regression (SVR) were studied. After a regression function is learned, we can predict the citation-to-go until 2011 for a paper given its features in 2002.

1) Results of Predicting for Old Papers: First we tested the predictors on the training papers. To recap what's going on, we've learned the predictors from the training papers (published before year 1990) with their observations between 1990 and 2002. Let's predict the citation count of these papers up to 2011.

To begin with, we observed that the pair-wise supervised learning is unstable for temporal prediction. In particular, even if the training error is small, the test error can blow up. Figure 3 (a) shows the linear regression fit of the past citation count until 2002, which looks reasonably good. However, using the model to predict the future citation count until 2012 is unstable, as shown in Figure 3 (b). It appears counter-intuitive at first sight. Since the correlation of the past citation count and future citation count is high (shown in Figure 3 (c)), one may expect to fit the future citation well by fitting a good model for the past citation count data. The catch is that the training input are the features in year 1990, but when predicting for the future, the input features are in year 2002. The weight vector learned by linear regression from the past data is expected to be used in combination with similar features to those used in training. However, the features for training and testing in the case of temporal prediction are very different, e.g., typically the magnitude of the features in 2002 is much bigger than those in 1990. So applying a paired-wise model learned from past citation count data to predicting future citation count is technically unsound. The case of SVR is similar, and thus figures are omitted here. This special difficulty is unique to temporal prediction especially because in the previously studied problem the input features for training and testing are in the same year. Figure 3 (d) shows the results of LSTD(0) for predicting the future citation count. Note that LSTD does not have the problem of the paired-wise supervised learning approach. The reason is that LSTD considers intermediate features, from which the algorithm actually extracts growth information (i.e., the linear model).

The effects of λ. In Figure 4, we compared the performance of LSTD(λ), where we employed L-2 regularization. We tuned the regularization factor (δ) and only showed those that performed best. Note that in comparing LSTD(λ) usually the same regularization factor is applied to LSTD(λ) for different λ. That is not a fair comparison, because λ influences the magnitude of entries of the LSTD matrix, which implies that different regularization has to be used for different λ. The result here confirms that the performance of LSTD(λ) greatly depends on the regularization factor. Generally different λ requires a different regularization factor for LSTD to perform the best. In particular, in this example, larger λ performs better

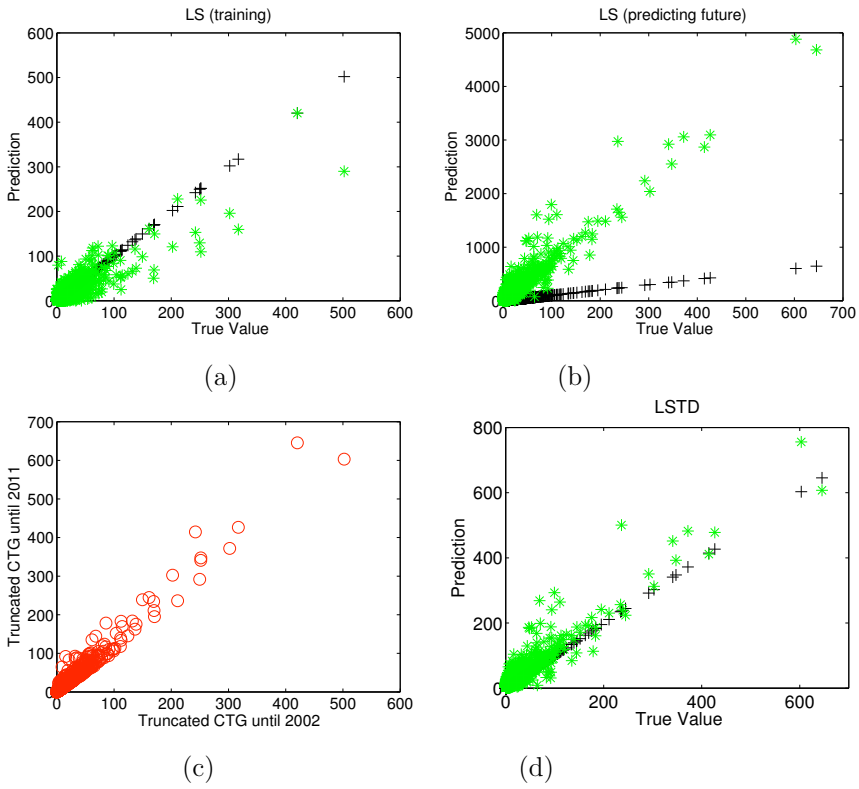

Fig. 3: (a): Linear regression model of the past citation count of the training papers. (b): The instability of the linear regression model for predicting future citation counts of the training papers. (c): The correlation between the past citation count and future citation count of the training papers. (d): LSTD(0) for predicting future citation counts of the training papers.

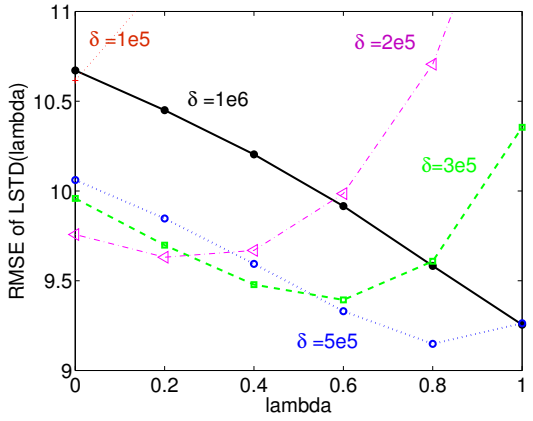

Fig. 4: The RMSE of LSTD(λ) for predicting the future citation count of the training papers.

when the regularization factor is well chosen. However, it is also noticeable that the larger λ is, the more sensitive LSTD(λ) is to the regularization factor. This means that in practice tuning the regularization factor is harder for LSTD with larger λ.

2) Results of Predicting for New Papers: That LSTD works well for predicting the future citation counts of the training papers shows that LSTD successfully generalizes over the time for this group of papers. A more challenging question is, *does LSTD generalize well to the other papers, especially to those newly published papers?* This requires that the algorithm generalizes in both papers and time. In fact, this is a fundamental question that a temporal machine learning predictor has to address. To study this question, we studied newer papers, in particular, the following three groups of papers, papers that are published between 1990 and 1995, between 1995 and 2000, and between 2000 and 2002. Note that these papers are obviously not in the training set.

Figure 5 shows the results of predicting for these groups of newly published papers using LSTD(0). The points marked with the cross are the true CTG, and the points marked with the star are the predictions. In addition, we mark papers published in different years with different colors to have a better resolution. In each group of papers, newer papers tend to have more citations, and predicting for them has a larger variation. The prediction is much more accurate for the first group of papers than for the other two groups of newer papers. This indicates that LSTD generalizes best to the papers that are published a few years after the beginning of the observation period (year 1990 in this case). Figure 6 summarizes the performance of LSTD(λ) with several typical regularization factors. Again the performance of LSTD(λ) depends greatly on the regularization factor. Unlike Figure 4 which shows that large λs perform the best in predicting the future citation of the training papers, Figure 6 shows that LSTD(λ) perform similarly for all λ in predicting for the newer papers when the regularization factor is well tuned. It is also noticeable that the

LSTD(λ) predictions are less accurate for newer papers for all λ.

C. Short-term Predictions

We also conducted experiments of predicting future k years of citation counts for both old and new papers. Figure 7 illustrates the performance of Dyna-based projection method. Firstly, as k increases the citation count is larger and has more variation, and prediction has larger errors. Note the conclusion here is different from those by [22], who discovered that in their problem predicting for the citation count in a larger time span is more accurate than for that in a smaller one. This illustrates a unique challenge in temporal prediction. Secondly, it can be observed that for large k the performance of the projection method gets closer to the LSTD algorithm. For one thing, this is because for large k the tails of the CTG become smaller. For the other, when k goes to infinity, Dyna method is theoretically equivalent to LSTD [14, 20] if one has a perfect model. However, in practice the model always has some errors like in this problem. We observed that Dyna(10) is much worse than LSTD. Besides the model error, that the magnitude of the reward signal being very large also contributes to the long-term prediction error of Dyna.

Finally, Figure 8 summarizes the performance of Dyna-based projection method for predicting k years of citation count data in the future, where $k = 1, 2, \ldots, 8$. For all values of k, across all groups of papers, predicting more years into the future has larger errors. For the same k, predicting for newer papers has larger errors. This is consistent with the long-term citation count prediction experiment. Similar to the case of long-term prediction, the pair-wise approach is unsound for making short-term temporal predictions.

D. Effects of Features

Table I summarizes the performance of various feature combinations for making $1, 5, 10$ years of predictions. The 10-year prediction was made by LSTD(0), and the other two were made by Dyna-based projection method. Generally the best performance is given by the combination of all the features for all prediction tasks. We started with the non-identity bibliometric features (authors, keywords and venues), then kept adding features to see if there is improvement. The results show that adding historical citation count features is most useful. Adding citation based features gives a slightly better performance than adding author identity features. The identity features are especially useful for predicting for old papers. When predicting for newer papers, however, the improvement of adding the identity features is smaller. This may be due to the fact that many of the authors of the old papers (used for training) do not appear in new papers.

VII. Conclusion

We studied the problem of predicting multi-step trends of citation counts given data only from the past. While we are not sure we are the first to study this practical problem, all the relevant work we found is limited to predicting the number of citations that already happened. In essence their predictors generalize only in papers published during the same period, but ours generalizes also in time. We showed that the pair-wise supervised learning approach, widely used by previous research for the former problem, is unstable for making temporal predictions. We generalize citation count to a general value function, and explore the use of reinforcement learning for temporal citation count predictions. In particular, we use LSTD for making long term predictions about citation count, and extend linear Dyna to do short-term predictions. Empirical results on DBLP show that both LSTD methods and Dyna-based method produce stable predictions. Some conclusions and observations are as follows. First, predicting citation count deeper into the future is more difficult because it has more uncertainty and variations. Second, the performance of LSTD(λ) depends greatly on L-2 regularization, and requires different regularization for different λ. LSTD(λ) perform similarly for all values of λ for newly published papers if the regularization factor is well tuned, though for predicting old papers LSTD with large λs perform better than those with small λs. Furthermore, the regularization factor for larger λ is harder to tune. Third, results also show that, for both short- and long-term predictions, it is less accurate for more recent papers. This should be due to the fact that more recent papers contain new information (content, authors, etc.).

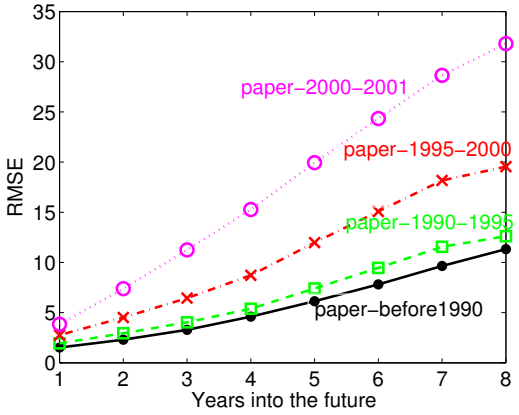

Fig. 8: Summary of the Dyna projection method for predicting short-term citation count.

Acknowledgement

We thank Dr. Jie Tang for making their DBLP collection available online. We thank Dr. Andras Gyorgy, Dr. Csaba Szepesvári, and other members of the RLAI group and Dr. Davood Rafiei's group for their insights.

References

[1] Agarwal, D., Chen, D., Lin, L.-J., Shanmugasundaram, J., Vee, E., 2010. Forecasting high-dimensional data. SIGMOD.

[2] Bertsekas, D. P., Tsitsiklis, J. N., 1996. Neuro-dynamic Programming. Athena.

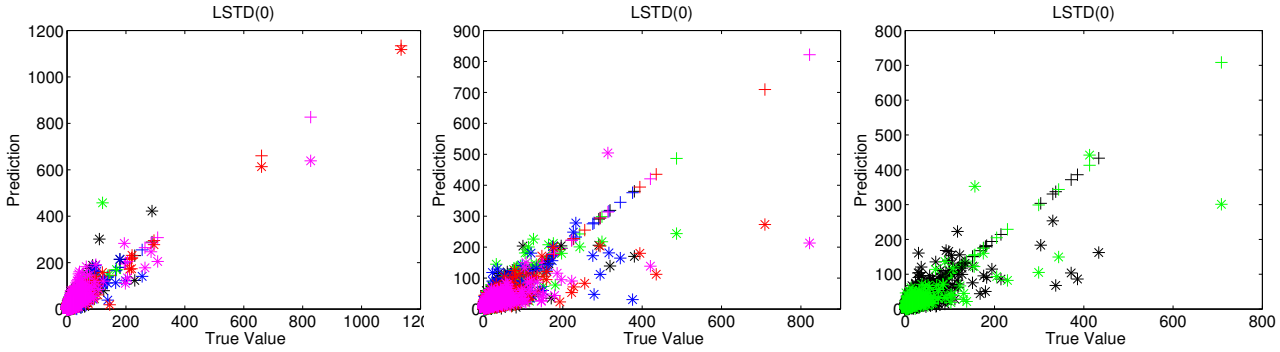

Fig. 5: LSTD(0) for predicting the CTG from 2002 to 2011 for newly published papers. Note that the LSTD algorithm is trained from data up to 2002 ("now"). In the three plots, the points with the cross marker correspond to the true CTG, and the other points (with the star marker) correspond to the predicted CTG. In the left plot, points in black, green, blue, red, magenta correspond to the papers published between 1990 and 1995. In the middle plot, these same colors correspond to the papers published between 1995 and 2000, respectively. In the right plot, the points in black and green correspond to the papers published in 2000 and those in 2001.

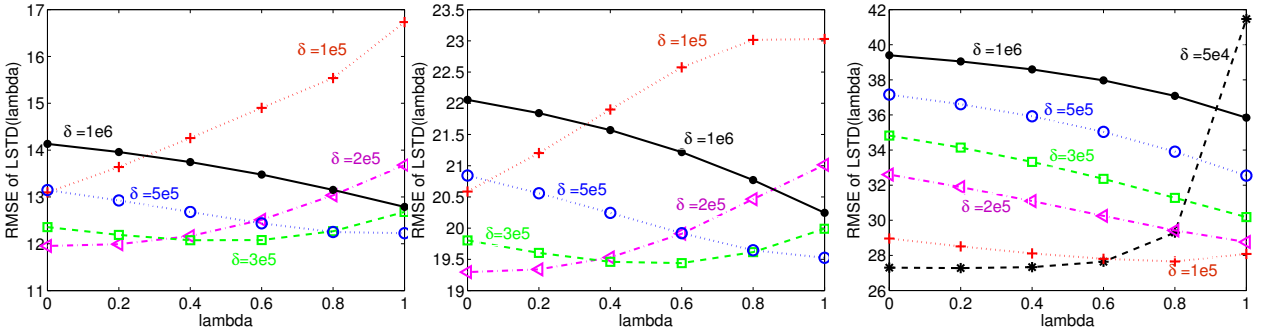

Fig. 6: Summary of LSTD(λ) for predicting the CTG from 2002 to 2011 for newly published papers. From left to right: LSTD(λ) for papers published between 1990 and 1995, papers between 1995 and 2000, and papers between 2000 and 2002.

[3] Boyan, J. A., 2002. Technical update: Least-squares temporal difference learning. Machine Learning 49, 233–246.

[4] Bradtke, S., Barto, A. G., 1996. Linear least-squares algorithms for temporal difference learning. Machine Learning 22, 33–57.

[5] Feitelson, D. G., Yovel, U., 2004. Predictive ranking of computer scientists using citeseer data. Journal of Documentation 60, 44–61.

[6] Fu, L. D., 2008. Improving biomedical information retrieval citation metrics using machine learning. Ph.D. thesis, Vanderbilt University.

[7] Fu, L. D., Aliferis, C., 2008. Models for predicting and explaining citation count of biomedical articles. AMIA Annu Symp Proc, 222–226.

[8] Garfield, E., 1955. Citation indexes for science: A new dimension in documentation through association of ideas. Science 122 (3159), 108–111.

[9] Gehrke, J., Ginsparg, P., Kleinberg, J., December 2003. Overview of the 2003 KDD cup. SIGKDD Explor. Newsl. 5, 149–151.

[10] Golbandi, N., Katzir, L., Koren, Y., Lempel, R., 2013. Expediting search trend detection via prediction of query counts. In: Proceedings of the sixth ACM international conference on Web search and data mining. WSDM '13. pp. 295–304.

[11] Ibáñez, A., Larrañaga, P., Bielza, C., Dec. 2009. Predicting citation count of bioinformatics papers within four years of publication. Bioinformatics 25 (24), 3303–3309.

[12] Ma, N., Guan, J., Zhao, Y., March 2008. Bringing pagerank to the citation analysis. Inf. Process. Manage. 44, 800–810.

[13] Modayil, J., White, A., Sutton, R. S., 2012. Multi-timescale nexting in a reinforcement learning robot. Proceedings of the International Conference on Adaptive Behaviour.

[14] Parr, R., Li, L., Taylor, G., Painter-Wakefiled, C., Littman, M. L., 2008. An analysis of linear models, linear value-function approximation, and feature selection for reinforcement learning. ICML.

[15] Sayyadi, H., Getoor, L., 2009. Futurerank: Ranking scientific articles by predicting their future pagerank. SDM, 533–544.

[16] Singh, S. P., Sutton, R. S., 1996. Reinforcement learning with replacing eligibility traces. Machine Learning 22 (1-3), 123–158.

[17] Sutton, R. S., 1988. Learning to predict by the methods of temporal differences. Machine Learning 3, 9–44.

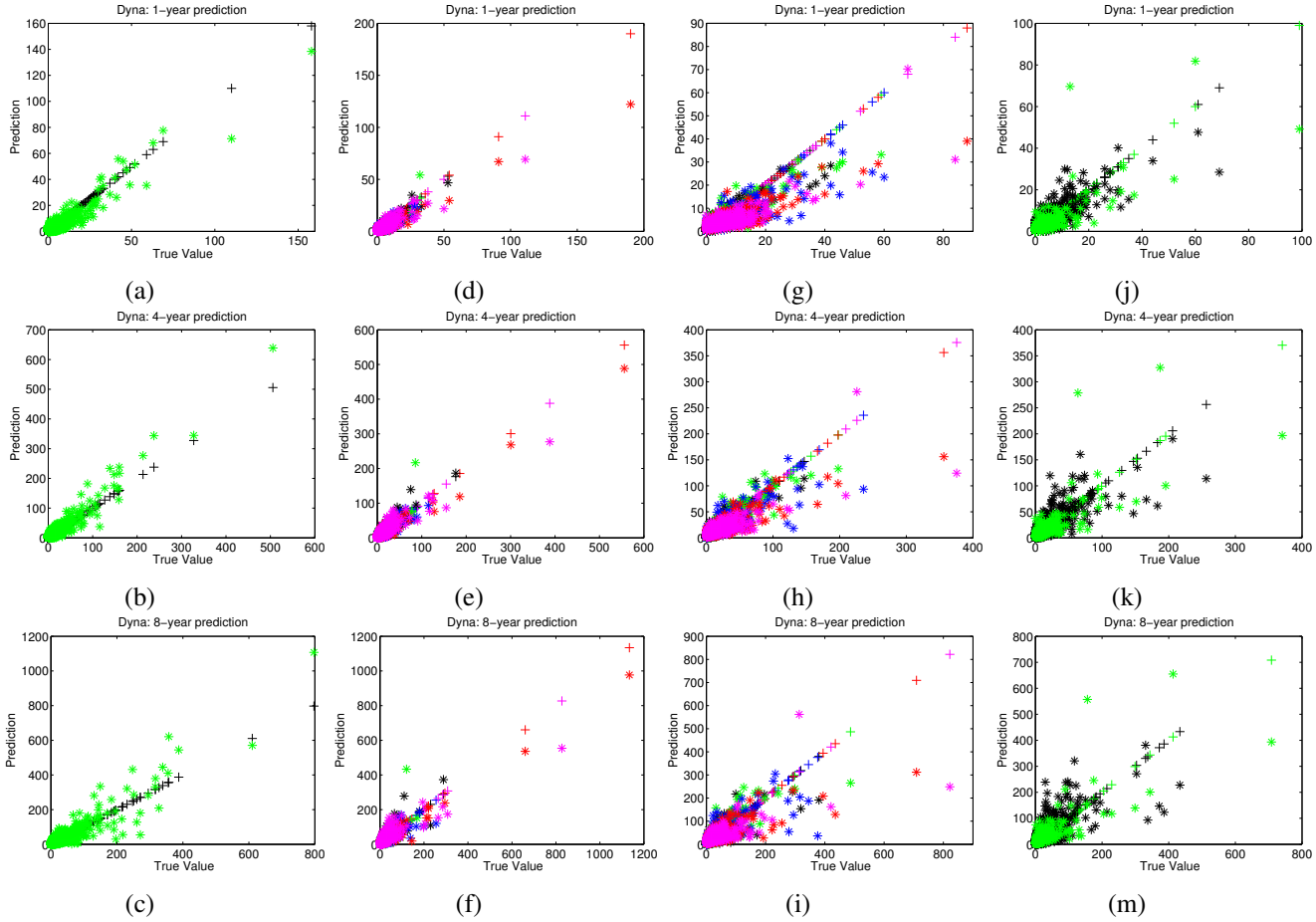

Fig. 7: Dyna-based projection method for predicting future citation counts of old and new papers. (a-c): old papers (published before 1990). (d-f): papers published between 1990 and 1995. (g-i): papers published between 1995 and 2000. (j-m): papers published in 2000 and 2001. Note that the (linear) Dyna model is learned from data up to 2002. Then we predict k years into the future from 2002 with Dyna, where here we show $k = 1, 4, 8$.

[18] Sutton, R. S., 1996. Generalization in reinforcement learning: Successful examples using sparse coars coding. NIPS.

[19] Sutton, R. S., Barto, A. G., 1998. Reinforcement Learning: An Introduction. MIT Press.

[20] Sutton, R. S., Szepesvári, C., Geramifard, A., Bowling, M., 2008. Dyna-style planning with linear function approximation and prioritized sweeping. UAI.

[21] Tang, J., Zhang, J., Jin, R., Yang, Z., Cai, K., Zhang, L., Su, Z., 2011. Topic level expertise search over heterogeneous networks. Mach. Learn. 82, 211–237.

[22] Yan, R., Tang, J., Liu, X., Shan, D., Li, X., 2011. Citation count prediction: Learning to estimate future citations for literature. CIKM.

TABLE I: Comparison of various features for making short-term and long-term predictions. b stands for using non-identity bibliometric features based on authors, keywords and venues. a stands for using identity-based author features. r stands for using historical reward (citation) features. c stands for using features based on the citing papers.

	Papers-before-90			Papers-90-95			Papers-95-00			Papers-00-02		
	1-year	5-year	10-year	1-year	5-year	10-year	1-year	5-year	10-year	1-year	5-year	10-year
b	3.9	16.1	26.4	4.0	16.6	27.1	4.9	20.4	29.7	6.4	29.7	40.9
$b+a$	3.6	14.9	23.8	3.8	15.7	25.7	4.8	20.2	29.7	6.4	29.7	40.9
$b+c$	3.3	14.6	23.5	3.7	15.1	25.7	4.7	19.3	30.6	6.1	28.5	39.1
$b+r$	1.4	6.8	11.0	1.8	7.5	12.0	2.5	11.6	19.2	4.0	19.2	27.3
$b+a+r$	1.4	6.1	10.6	1.8	7.3	11.9	2.5	11.6	19.2	3.8	18.7	27.3
$b+c+r$	1.4	6.0	10.3	1.8	7.1	11.7	2.2	11.7	19.0	3.6	18.3	27.2
$b+a+c+r$	1.4	6.1	10.0	1.8	7.1	11.2	2.1	11.3	19.1	3.6	18.2	27.0