A FRAMEWORK FOR EXPLOITING TEMPORAL VARIATIONS IN RELATIONAL DOMAINS

A Thesis

Submitted to the Faculty

of

Purdue University

by

Umang Sharan

In Partial Fulfillment of the

Requirements for the Degree

of

Master of Science

August 2008

Purdue University

West Lafayette, Indiana

To my family for their advice and support over the years.

ACKNOWLEDGMENTS

Many thanks to Prof. Jennifer Neville for being an ideal thesis supervisor. I am grateful for her pertinent guidance and encouragement that helped immeasureably during the preparation of this thesis. I am also thankful to my committee, Prof. Guy Lebanon and Prof. Luo Si.

Special thanks also to my group members Pelin Angin, Hoda Eldardiry and Rongjing Xiang for our invaluable group meetings and insightful discussions.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

Sharan, Umang. M.S., Purdue University, August, 2008. A Framework for Exploiting Temporal Variations in Relational Domains . Major Professor: Jennifer Neville.

Many relational domains contain temporal information and dynamics that are important to model. As an example, consider scientific publication networks—paper publication events occur over time and coauthor relationships form and develop over time. The temporal aspects of such data can reveal relevant relationships and indicate relationship strength.

In this work, we focus on incorporating temporally-varying relationship information in predictive models of attributes. By analyzing the temporal dynamics, we aim to identify and emphasize more *influential* relationships, thus improving the performance of models that consider the characteristics of related entities during prediction. We present a framework that models dynamic relational data with a two-phase process, first summarizing the temporal-relational information with kernel smoothing, and then moderating attribute dependencies with the summarized relational information. We evaluate our approach on three real-world datasets and show that it results in significant performance gains compared to two baseline approaches that ignore the temporal aspects of the data.

1 INTRODUCTION

Recent research has demonstrated the utility of modeling relational information for domains such as web analytics [1], marketing [2] and fraud detection [3]. This work has demonstrated that incorporating the characteristics of *related* instances into statistical models improves the accuracy of attribute predictions. One key reason for these improvements is the presence of *autocorrelation* or *homophily* in relational data. Homophily refers to the tendency of like to associate with like [4] and autocorrelation refers to correlation of the values of an attribute between pairs of related instances [5]. For example, fraud exhibits autocorrelation—if we know one person is involved in fraudulent activity, then his associates have increased likelihood of being engaged in misconduct as well [6]. Similarly, webpage topics are often autocorrelated, which means that two hyperlinked pages are more likely to share the same topics than two randomly selected web pages [7]. The presence of homophily and autocorrelation offers a unique opportunity for relational learning techniques to improve model performance because the similarity among related instances can be exploited by the model to improve predictions.

A number of techniques have been developed to successfully exploit dependencies in relational domains (see e.g., [8]), but this work has focused primarily on modeling *static* relational data. For the numerous relational domains that have temporal dynamics, researchers have generally analyzed static *snapshots* of the data, which consist of all the objects, links, and attributes that have occurred up to and including time t [1–3]. This approach ignores the temporal information present in the data and limits the applicability of the models. In many datasets there are likely to be dependencies between the temporal and relational information that can be exploited to improve model performance. For example, in fraud detection a pair of individuals that communicate regularly over time may be more likely to exhibit autocorrelation

than a pair of individuals that communicate heavily but only for a brief time period. Similarly, in marketing domains customers that have recently purchased a product may be more likely to advertise that product to their friends than customers that purchased the product in the more distant past. To date, there are few available data mining tools that can simultaneously exploit temporal and relational dependencies in the data.

Relational data may exhibit temporal dynamics in a number of dimensions. First, the instances in the data may appear and disappear over time. For example, web pages are created as web sites are developed, expanded, and modified over time. It may be important to model instance age if recently added instances exhibit different characteristics than older instances (e.g., new vs. established accounts). Second, the links (or relations) in the data may represent events at a particular time. If this is the case, then the time associated with the event may be important to model (e.g., the publication date of a scientific paper). In addition, multiple links between two instances may occur over time. If this is the case, properties of the sequence of links may be important to model (e.g., how often a pair of coauthors publish together). Third, the attribute values in the data may change over time. For example, a sensor may be recording the position of an object moving through a building and this may inform predictions about the properties of the object.

Recent work in statistical relational learning has only just begun to investigate these temporal dimensions. Some initial work has focused on transforming temporal-varying links and objects into aggregated features [3] and other work has focused on modeling the temporal dynamics of time-varying attributes [9]. There have been some efforts to model temporally-varying links to improve automatic discovery of relational communities or groups [10, 11] but this work has not attempted to exploit the temporal information in a classification context.

The goal of this work is to improve attribute prediction in dynamic domains by incorporating time-varying links into statistical relational models. One motivation for modeling time-varying links is the identification of influential relationships in the data.

Since much of the success of relational models is predicated on the correlation between attribute values of linked instances, a method that prunes away spurious relationships and highlights stronger relationships will lead to more significant increases in model performance.

We conjecture that the temporal link information will be useful for disambiguating relationships in this fashion. In particular, we look for patterns of *temporal locality* and *temporal recurrence* to identify the relationships that are more likely to exhibit autocorrelation dependencies. Temporal locality refers to the notion that events in the recent past are more influential than events in the distant past. Temporal recurrence refers to the notion that a regular series of events between two instances is more likely to indicate a stronger underlying relationship than an event isolated in time.

As illustration, consider the following example from the Internet Movie Database (IMDb; www.imdb.com). Table 1.1 lists the set of movies that actors Owen Wilson and Ben Stiller have co-starred in the last seven years and the gross earnings (adjusted for inflation) of those movies. This example indicates that recent successful relationships—'Meet the Fockers' and 'Starsky & Hutch'—outweigh less successful relationships in the past—'Zoolander' and 'The Royal Tenenbaums'—for predicting the success of 'Night at the Museum'.

Table 1.1
Example temporal pattern in IMDb

Movie	**Release Year**	**Earnings ($ million)**
Zoolander	2001	54
The Royal Tenenbaums	2001	62
Starsky & Hutch	2004	99
Meet the Fockers	2004	315
Night at the Museum	2006	264

As another example, consider the Cora database of computer science research papers extracted automatically from the web [12]. Each paper has an associated topic and citations relating it to other papers that have been published in the past. Figure 1.1 shows the autocorrelation between the topics of papers published in the year 1996 with the topics of the papers they cite in the past. The x-axis represents the time interval between 1996 and the year of publication of the cited papers. Observe that the correlation between topics decreases as the time lag increases. Thus, the topics of recent references are likely to be better indicators than the topics of references that were published farther in the past.

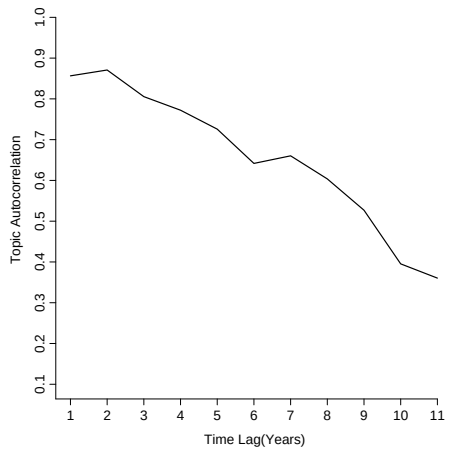


Figure 1.1. Temporal variation of topic autocorrelation in Cora

In this work, our aim is to analyze the temporal sequences of interactions between pairs of instances to identify the nature of their relationships and then incorporate this information to moderate the influence of related instances in predictive models of attributes. To do this, we propose the Time Varying Relational Classifier (TVRC) [13, 14], a novel framework for incorporating time-varying link information into statistical relational models. TVRC uses a two-step process that first transforms a dynamic relational graph into a static weighted summary graph using kernel smoothing. The second phase then incorporates the static link weights into a modified relational classifier to moderate the influence of attributes throughout the relational

data graph. We evaluate our approach on three real-world datasets and show that it achieves significant improvements in accuracy compared to two baseline models that ignore the temporal dimension of the data.

The remainder of this thesis is organized as follows: Section 2 outlines the background and related work in statistical learning in temporal relational domains. Section 3 discusses the details of the TVRC framework and its implementation. Section 4 describes the experimental evaluation results. Section 5 concludes and suggests some possible future research directions.

2 BACKGROUND

This work focuses on statistical learning in temporal relational domains. Many relational domains like fraud detection, web analysis and bioinformatics have temporal variations over time. There are two aspects to temporally changing relational data—temporally varying attributes and temporally varying link structure.

Temporal changes in attributes have been exploited in the past in non-relational contexts to improve prediction accuracies in time series models. Autoregressive models exploit temporal autocorrelation to predict the value of a variable X at time t based on the value of X at previous time steps [15–17]. These models are often used in econometrics to model time-varying data such as stock prices and interest rates (see [18]). An autoregressive model $AR(\rho)$ of order ρ is defined as:

$$X_t = c + \sum_{i=1}^{\rho} \beta_i X_{t-i} + \epsilon_t$$

where $\beta_1, \cdots, \beta_\rho$ are the parameters of the model, c is a constant and ϵ_t is a error term. Autoregressive models typically represent dependencies by including a lagged value of the response variable as a regressor. In other words, they model the variable at the current time step as a parameterized function of the past.

An alternative way to model temporally changing attributes is to model the variables explicitly as a time-series whereby each observation in the series depends on a hidden state which is correlated over time. Sequential linear series models like Hidden Markov Models (HMM) define a probability distribution over sequences as follows:

$$P\{X_t, Y_t\} = P(X_1)P(Y_1|X_1)\prod_{t=2}^{T} P(X_t|X_{t-1})P(Y_t|X_t)$$

where $\{Y_t\}$ are the observed variables and $\{X_t\}$ are the corresponding hidden state variables. Figure 2.1 shows the graphical representation of such a sequential model

and the dependencies among the X_t. The dependencies follow a first-order Markov assumption where X_t depends only on the state X_{t-1} at the previous time step. HMMs and other sequential models are used for a variety of applications in speech recognition, filtering and control applications, and protein sequence recognition. Recent work in Dynamic Bayesian Networks (DBN) [19, 20] has extended these sequential models to use arbitrary Bayesian networks at each time step. However, exact probabilistic inference is generally intractable in DBNs and approximate inference methods have to be used instead.

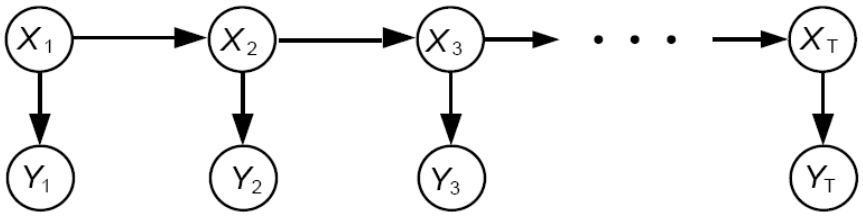


Figure 2.1. An example sequential Bayesian network.

Temporal changes in link structure have been modeled by Cortes et al. [6, 21] and Hill et al. [11, 22]. This work has used Communities of Interest (COI) to summarize and represent temporally varying link structure in a novel way to facilitate large-scale analysis of dynamic networks such as telecommunications call networks. It addressed the following major challenge—to process massive volumes of data efficiently while accounting for the dynamic nature of transactional data by capturing the most relevant information and eliminating less useful and spurious information. The COI representation captures relational changes in a concise way that evolves smoothly through time. Cortes et al. [6, 21] use this representation for fraud detection analysis in telecommunication networks.

Another approach towards modeling time-varying relational networks has used more complex sequential linear models [23, 24]. This work has modeled temporal sequences of network structures as first order Markov chains whereby each network instance is assumed to be generated by an Exponential Random Graph Model

(ERGM) [25]. The probability distribution function for a sequence of network instances $N = \{G_1, G_2, \cdots, G_T\}$ is defined as:

$$P(N) = \frac{1}{Z(\theta)} exp\{\theta' \psi(N)\}$$

Using a first-order Markov assumption that the graph at time t only depends on the graph at time $(t-1)$, we can simplify the model as follows:

$$P(G^t|G^{t-1}) = \frac{1}{Z(\theta, G^{t-1})} exp\{\theta' \psi(G^t, G^{t-1})\}$$

MCMC estimation techniques have been developed to learn the parameters of these models but this approach is not tractable for large datasets or for models with higher-order Markov assumptions. Further, this model is designed for link prediction tasks and is not geared for attribute prediction tasks.

In the area of statistical relational learning, some recent work has focused on modeling both attributes and links changing over time. Sanghai et al. [9] proposed the Dynamic Probabilistic Relational Model (DPRM) for modeling temporal relational domains. DPRMs combine Probabilistic Relational Models (PRM) [26–28] with DBNs [19]. In essence, DPRMs are a series of PRMs for each time step linked together through temporal slot chains following the first-order Markov assumption. For any time window T, the joint distribution over the instantiations of the relational schema I at different time steps $(I_0, ..., I_T)$ is given by:

$$P(I_0, ..., I_T) = P_0(I_0) \prod_{t=1}^{T} P(I_t|I_{t-1})$$

DPRMs are computationally very intensive for both learning and inference due to the large space of possible dependencies. In addition, DPRMs use a restrictive first-order Markov assumption which may not be sufficient to capture the notions of *temporal locality* and *temporal recurrence* that we focus on in this work.

McGovern et al. [29] recently proposed Spatio-Temporal Relational Probability Trees (SRPTs) as an extension to Relational Probability Trees (RPTs) [30] to incorporate temporal variations in relational attributes and links. SRPTs are designed for

relational domains where concepts vary based on small spatial and temporal scales within the data (e.g., weather prediction). They model the temporal relational information by expanding the set of features explored in the learning algorithm. For example, the feature set includes *Temporal Exists*, which assesses whether an object or a link lasted at least t time steps, and *Relative Count*, which splits data on the relative change in the number of matching items (count) within a time window. Thus, SRPTs are able to model some aspects of time-varying link structure (i.e., local degree changes) as well as time-varying attributes in relational data. Although adding temporal components to the feature space of relational decision trees provides the flexibility to model temporal variations in both attributes and relationships, it results in an exponential number of possible features which may be infeasible to explore for large datasets. McGovern et al. make this tractable by defining a restricted set of temporal relational features manually, based on domain knowledge specific to weather prediction tasks.

In this work, we build on the COI [6] representation to capture temporal changes in relational structure to improve attribute prediction in domains where the link structure is varying with time. Our approach uses the temporal information in a novel way to improve attribute prediction in relational domains. We move beyond the restrictive Markov assumptions of the past work and focus on summarizing the temporal information before incorporating it into the models. This facilitates reasoning about patterns of temporal locality and temporal recurrence that we believe are crucial for identifying and exploiting influential relationships in the data.

3 TVRC FRAMEWORK

We consider relational data represented as an attributed multi-graph—more specifically, as a directed, attributed graph $G = (V, E)$, with V (nodes) representing objects and E (edges) representing relations[1], having one or more connected components. To elaborate, the nodes V represent objects in the data (e.g., people, organizations, events) and the edges E represent relations among the objects (e.g., works-for, located-at). Each node $v_i \in V$ and edge $e_j \in E$ are associated with a type $G(v_i) = g_{v_i}$, $G(e_j) = g_{e_j}$. Each object or link type $g \in G$ has a number of associated attributes $\mathbf{X^g} = (X_1^g, ..., X_m^g)$ (e.g., age, gender).

When relational data has a temporal component, there are three aspects of the data that may vary over time. First, attribute values may vary over time $\mathbf{X_i} = (X_{i_1}, X_{i_2}, ..., X_{i_t})$. Second, relationships may vary over time. This results in a different data graph $G_t = (V, E_t)$ for each time step t, where the nodes remain constant but the edge set may vary (i.e., $E_{t_i} \neq E_{t_j}$ for some i, j). Third, objects existence may vary over time (i.e., objects may be added or deleted). This is also represented as a set of data graphs $G'_t = (V_t, E_t)$, but in this case both the objects and links sets may vary.

Initial efforts to incorporate time into statistical relational models have focused on the first and third cases. For domains where attributes vary over time, Sanghai et. al [9] have extended probabilistic relational models (PRMs) [26] in a manner similar to dynamic Bayes networks [20], rolling out a separate PRM for each time step and modeling the dependencies of attribute values in one time step to the next. For domains where the number of objects is uncertain and can vary, Milch et. al [31] have developed a Bayesian logic modeling approach to reason about possible worlds

[1]There may be more than one edge between a pair of nodes.

with varying numbers of objects. To our knowledge, there are no statistical relational models that exploit temporal information in domains where the links change over time.

In this work, we focus on the case where the links vary over time and outline a framework for such domains. For example, consider the case where we have a set of people coauthoring scientific papers. In each year, there will be different sets of people coauthoring different papers. This is represented in Figure 3.1. The nodes (authors) remain constant but the edges change in each time step, representing the coauthor and citation events that have occurred in each year.

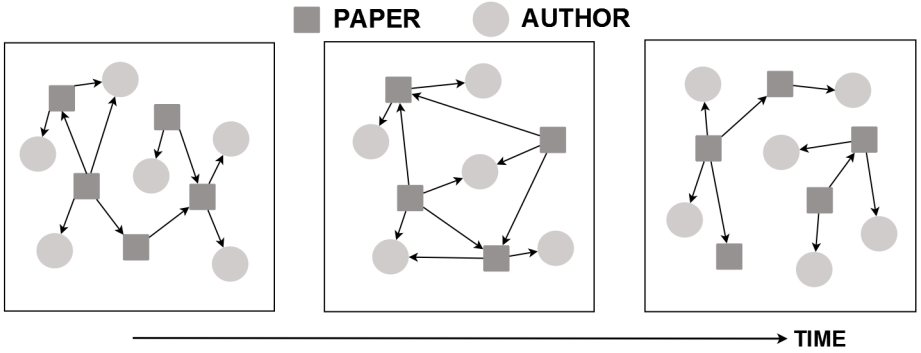


Figure 3.1. Temporally varying relationships over time.

Our approach, the Time Varying Relational Classifier (TVRC), is a two-phase prediction framework, which consists of a *graph summarization* phase and a *relational classification* phase. The key idea of TVRC is to exploit the temporal influence of relationships across snapshots by summarizing them suitably and utilizing the summarized values in a modified relational classification model. Figure 3.2 illustrates the general framework of TVRC approach. Consider a dataset of research papers with reference and authorship information. The summarization phase in Figure 3.2 summarizes different snapshots of the data which correspond to the papers published in different years. The relational model used in the classification phase exploits the edge weights w_i learnt during the graph summarization phase to produce more accurate predictions. We now describe each phase in more detail.

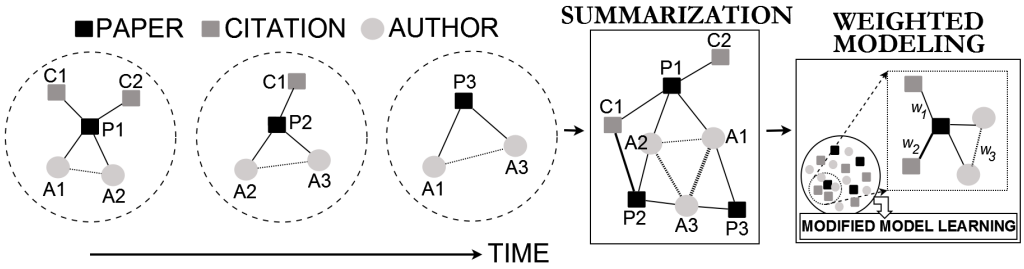


Figure 3.2. General framework for TVRC.

3.1 Graph Summarization

The graph summarization phase summarizes a temporal sequence of relational graphs into a weighted summary graph. Let $G_t = (V_t, E_t, W_t)$ be the relational graph at time step t, where $W_t = \{w_{ij} = 1 : e_{ij} \in E_t\}$ is a set of unit weights on the edges of G_t. Let $\{G_1, G_2, \cdots, G_n\}$ be the temporal snapshots of the data at consecutive time steps t, from $t = 1$ through n. Each temporal snapshot G_t contains the relationships between the objects in the dataset at time t. New objects and links may have been added or deleted from G_{t-1} to G_t. Figure 3.1 shows three different temporal snapshots depicting the varying relationships among a set of authors based on the papers they published in each time step.

We define the summary graph $G_t^S = (V_t^S, E_t^S, W_t^S)$ at time t as a weighted sum of the snapshot graphs $\{G_1, G_2, \cdots, G_t\}$

$$V_t^S = V_1 \cup V_2 \cup \cdots \cup V_t$$

$$E_t^S = E_1 \cup E_2 \cup \cdots \cup E_t$$

$$W_t^S = \alpha_1 W_1 + \alpha_2 W_2 + \cdots + \alpha_t W_t$$

where V_t and E_t are the vertex and edge sets respectively of the temporal snapshot G_t and W_t are the unit weights associated with the edges of G_t. The α's are weights which

determine the contribution of each temporal snapshot in the summary graph. More specifically, we formulate the graph summarization as a kernel smoothing operation:

$$W_t^S = \sum_{i=1}^{t} K(G_i; t, \theta)$$

where K is an appropriate kernel function with parameter θ. Representing the summary operation through kernel smoothing gives us the freedom to explore and choose a suitable weighing scheme from a wide range of kernel functions, so as to best capture and exploit the temporal variations in the subsequent classification phase.

The graph summarization phase of TVRC depends on the smoothing kernel applied on the snapshot graphs. From Figure 1.1, it is clear that past information loses its influence on the present attributes rapidly. Thus, we postulate that a paper's topic is more likely to be influenced by the topic of a reference that has been published recently rather than one that was published farther in the past. In addition, we postulate that an author is more likely to currently be working on a topic similar to those of his recent coauthors than those of his past coauthors. To exploit these ideas of temporal locality and temporal recurrence, we employ a weighting scheme based on decaying kernels. We explore the following three kernels for summarization—exponential kernel, linear kernel, inverse linear kernel.

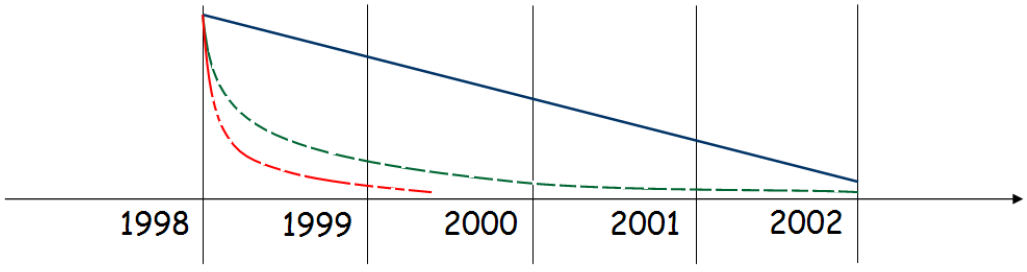


Figure 3.3. Example decay of a link event that occurs in 1998. The solid curve shows the linear kernel, the uniformly dashed curve shows the inverse linear kernel, while the non-uniformly dashed curve shows the exponential kernel.

3.1.1 Exponential Kernel

The exponential kernel weighting scheme is similar to the approach used by Cortes et al. [6] for graph summarization. The kernel function K_E is defined as:

$$K_E(G_i; t, \theta) = (1-\theta)^{t-i}\theta W_i$$

The exponential kernel weights the recent past highly and decays the weight rapidly as time passes. The kernel smoothing operation on the input temporal sequence $\{G_1, G_2, \cdots, G_t\}$ can also be expressed as a recursive computation on the weights $\{W_1, W_2, \cdots, W_t\}$ through time. This means the summary graph at time t can be written as a weighted sum of the graph at time t and the summary graph at time $(t-1)$ where the summary parameter $\theta \in [0, 1]$ specifies the influence of the current time step and t_o is defined as the initial time step in the time window.

$$W_t^S = \begin{cases} (1-\theta)W_{t-1}^S + \theta W_t & \text{if } t > t_o \\ \theta W_t & \text{if } t = t_o \end{cases}$$

Figure 3.3 shows an example of how quickly the unit weight for an event in 1998 decays over time in the summary graph.

3.1.2 Linear Kernel

The linear kernel K_L is defined as:

$$K_L(G_i; t, \theta) = \theta W_i(\frac{t_* - t_i + 1}{t_* - t_o + 1})$$

where t_* is defined as the final time step considered in the time window. The linear kernel decays more gently (see Figure 3.3) and retains the historical information longer. Again, the summary graph at time t is the weighted sum of the graph at time t and the summary graph at time $(t-1)$, and the summary parameter $\theta \in [0, 1]$ and is defined as:

$$W_t^S = \begin{cases} (\frac{t_*-t}{t_*-t+1})W_{t-1}^S + \theta W_t & \text{if } t > t_o \\ \theta W_t & \text{if } t = t_o \end{cases}$$

3.1.3 Inverse Linear Kernel

The inverse linear kernel K_{IL} lies between the exponential and linear kernels when moderating the contribution of historical information and is defined as:

$$K_{IL}(G_i; t, \theta) = \theta W_i(\frac{1}{t_i - t_o + 1})$$

while the weights if the summary graph are recursively defined as:

$$W_t^S = \begin{cases} (\frac{t-t_o}{t-t_o+1})W_{t-1}^S + \theta W_t & \text{if } t > t_o \\ \theta W_t & \text{if } t = t_o \end{cases}$$

Regardless of the kernel employed, the above set of equations recursively define the summary weights W_t^S at time t as the weighted average of the summary weights W_{t-1}^S at time $(t-1)$ and the weights W_t at time t.

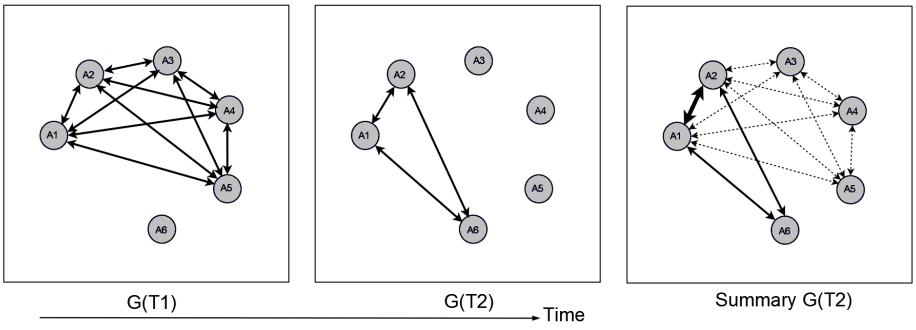


Figure 3.4. Example of graph summarization with kernel smoothing.

Figure 3.4 shows an example summarization of relationships between authors publishing together papers over two years. The dotted edges in the summary graph denote relationships weakened over time, thick solid edges denote frequent relationships that are strengthened over time, while thin solid edges denote relationships freshly occurring in the present time step. Note that our approach is general enough to handle cases when more than one link exists between a pair of nodes at a given time step. However, it does assume that if there are multiple links, they are all of the same type and thus can be summarized into a single relationship. For example, if two authors A

and B publish two papers and administer a grant together in year t, there will be one summary edge between A and B even though the coauthor links and the co-PI links do not necessarily indicate the same type of relationship between A and B. This case can be easily dealt with by partitioning the graph into a separate graph for each link type g before summarization.

3.2 Weighted Relational Classification

The second phase of TVRC algorithm is the Relational Classification Phase. Once we have summarized the relational data, we learn a predictive model on the summarized data, exploiting the temporal variations to improve the prediction results. In this work, we extend the Relational Bayes Classifier (RBC) [32] and the Relational Probability Tree (RPT) [30] to incorporate link weights from the summary graph into the models. Any relational model that can be extended to use weighted instances is suitable for this phase. In this work, we chose to focus on RBCs and RPTs because of their simplicity and efficiency.

3.2.1 Weighted Relational Bayes Classifier

RBCs extend naive Bayes classifiers to relational settings by treating heterogeneous relational subgraphs as a homogenous set of attribute multisets. For example, when modeling the dependencies between the topic of a paper and the topics of its references, the topics of those references form multisets of varying size (e.g., {NN, GA}, {NN, NN, RL, NN, GA}). The RBC models these heterogenous multisets by assuming that each value of the multiset is independently drawn from the same multinomial distribution. This approach is designed to mirror the independence assumption of the naive Bayesian classifier [33]. In addition to the conventional assumption of attribute independence, the RBC also assumes attribute value independence within each multiset.

More formally, for a class label C, attributes $\mathbf{X}$, and related items R, the RBC calculates the probability of C for an item i of type $G(i)$ as follows:

$$P(C^i|\mathbf{X},R) \propto \prod_{X_m \in \mathbf{X^{G(i)}}} P(X_m^i|C) \cdot \prod_{j \in R} \prod_{X_k \in \mathbf{X^{G(j)}}} P(X_k^j|C) \cdot P(C) \tag{3.1}$$

In equation 3.1, each attribute value is given an implicit weight of 1. In order to incorporate the weights from the summary graph we modify this equation to define the weighted RBC as follows:

$$P(C_t^i|\mathbf{X},R) \propto \prod_{X_m \in \mathbf{X^{G(i)}}} P(X_m^i|C) \cdot \prod_{j \in R} \prod_{X_k \in \mathbf{X^{G(j)}}} w_{ij}^t \cdot P(X_k^j|C) \cdot P(C) \tag{3.2}$$

where w_{ij}^t is the product of the weights in the summary graph G_t^S on the path from the target node i to the related node j. Figure 3.5 shows how the weighted RBC incorporates edge weights into its multiset representations.

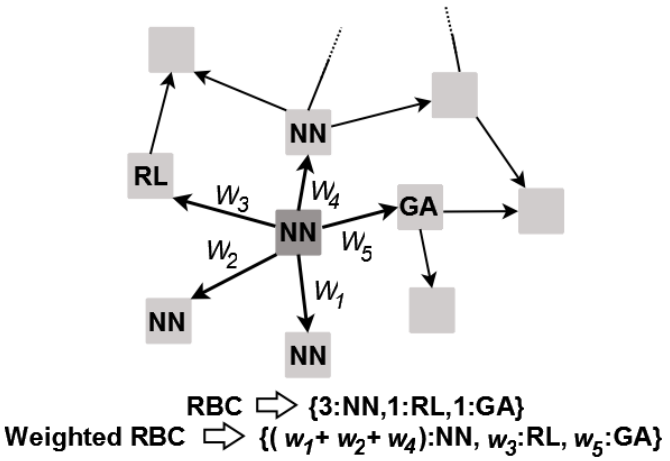


Figure 3.5. Modification to RBC multisets to include summary weights.

3.2.2 Weighted Relational Probability Trees

RPTs extend standard probability estimation trees to a relational setting in which data instances are heterogeneous and interdependent. The algorithm for learning the structure and parameters of a RPT searches over a space of relational features that

use aggregation functions (e.g. AVERAGE, MODE, COUNT) to dynamically propositionalize relational data multisets and create binary splits within the RPT. Similar to RBCs, each attribute value is implicitly given a weight of 1 in the RPT feature construction and evaluation phase. In order to incorporate the weights from the summary graph, we define a weighted RPT by modifying the counts of the attribute value when computing the RPT aggregate functions. Figure 3.6 illustrates how weighted RPT feature calculation differs from the standard RPT—in particular, while calculating the mode of the linked topics, the weighted RPT may end up calculating a different feature value based on the link weights in the multiset.

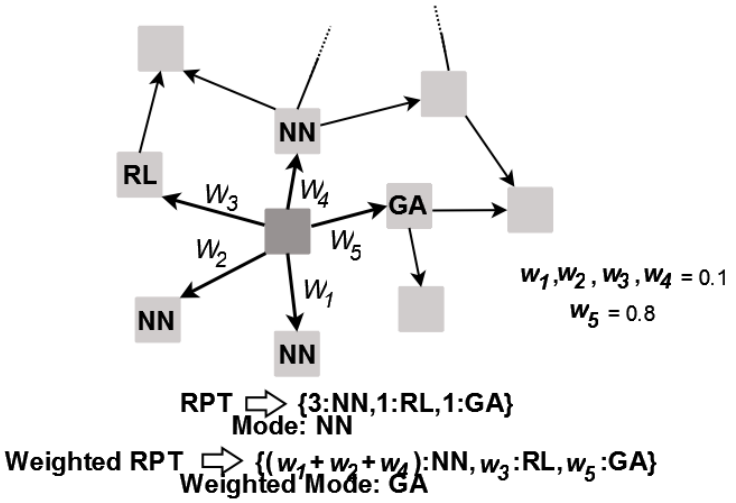


Figure 3.6. Modification to RPT feature calculation to include summary weights.

In both the RBC and the RPT, the weights can be viewed as probabilities that a particular relationship in the data is still active at the current time step t, given that the link was observed at time $(t - k)$. For exponential kernel, this corresponds to a geometric distribution with parameter θ, which specifies the probability that a link occurred k time steps in the past.

3.3 Learning

The weighted RBC uses standard maximum likelihood learning with Laplace correction for zero-values. More specifically, the sufficient statistics for each conditional probability distribution are computed as weighted sums of counts, based on the edge weights from the summarization phase. The weighted RPT uses the standard RPT learning algorithm except that the aggregate functions are computed after weighing each attribute value count with the corresponding edge weight learnt from the summarization phase.

We use k-fold cross validation to determine the summary parameter θ to be used during the graph summarization phase. We divide the training sample into k folds by sampling the instances independently. We then learn a TVRC by training it on $(k-1)$ folds and applying the model on the remaining fold using a range of values of θ. In the experiments below, we consider 10 values of θ in the set $\{0.1, 0.2, ..., 1.0\}$ and k=10. The θ value that achieves the maximum cross-validated accuracy on each fold is selected and the average of these values is used as the final θ value.

The time complexity of learning the RBC and RPT models is $O((t+m)E)$, where t is the number of time steps, m is the number of attributes used for prediction, and E is the number of edges in the summary graph G_t^S. The time complexity for the summary graph computation is $O(tE)$ since each edge must be considered in each summary calculation. In the worst case, when the graph is close to completely connected, the number of edges will be $O(V^2)$. However, in many relational datasets, node degree is bounded by a constant (i.e., a node's neighbors do not grow as a function of dataset size), thus $E << V^2$ in practice. The time complexity for the RBC learning algorithm as well as the RPT learning algorithm is $O(mV)$ where V is the number of target nodes (i.e., instances being predicted) assuming that the number of neighbors for each node (used to get a particular set of attribute values) can be bounded by a constant.

3.4 Prediction

For prediction, the TVRC computes the summary graph G_t^S at time t—the time step at which the model is being applied. Then we apply the model learned for time $(t-1)$ to G_t^S. The prediction phase of RBCs and RPTs is appropriately augmented to incorporate the link weights W_t^S from G_t^S.

4 EXPERIMENTS

We report the results of TVRC on three real world datasets. The experiments reported below evaluate the performance and prediction accuracy of TVRC against baseline models that ignore the temporal component of the snapshot data. The rest of the section is organized as follows: First, we describe the different datasets used for empirical evaluation of TVRC. Thereafter, we define the different models compared in the experiments, and finally we present an analysis comparing the performance of different models based on the results of the experiments.

4.1 Data

We considered three real world datasets for our experiments. The Internet Movie Database (IMDb) contains movie release information, including their earnings, actors, studios, directors, etc. The Cora database contains authorship and citation information about computer science research papers extracted automatically from the web. The Reality Mining database contains telephone call and mobile device proximity records amongst a set of students, faculty, and researchers at MIT. Table 4.1 lists the number of objects and/or links present in each dataset. We describe each dataset in more detail below.

4.1.1 IMDb

The Internet Movie Database (IMDb) is one of the largest publicly available databases of information about movies, actors, directors, studios, etc. The database is available as compressed text files at *http://imdb.com/interfaces*.

Table 4.1
Datasets used for empirical evaluation.

IMDb	**CORA**	**Reality Mining**
Movies: 5,301	Papers:16,153	People: 97
Actors: 126,641	References: 29,603	Devices: 20,795
Producers: 11,973	Authors: 21,976	Telephone Call
Studios: 391		Edges: 443,553
Directors: 2,535		Device Proximity
Editors: 2,186		Edges: 285,512
Cinematgrs: 1,518		
Time Window:	Time Window:	Time Window:
1981-2007	1981-1998	May-Nov 2004

For this work, we selected the set of movies that were released in the time period 1981-2007. We did not consider any movies from the genres *Television* or *Adult*. Each movie has an attribute 'Gross earnings', which records the amount of money the movie grossed in total. We adjusted these values to account for inflation and make the values comparable across different years. Furthermore, we only considered movies with (adjusted) gross earnings $> \$1mil$.

Table 4.2
Example aliases for Columbia TriStar.

Movie Name	**Studio**
Between Brothers(1999)	($\surd$) Columbia TriStar Television [us]
Channel Umptee-3(1997)	($\surd$) Columbia TriStar Children's Television [us]
Fun Factory, The(1976)	($\surd$) Columbia Pictures Television [us]
Shipmates(2001)	($\surd$) Columbia TriStar Domestic Television [us]
Jeremiah(2002)	(X) Province of British Columbia Production ...

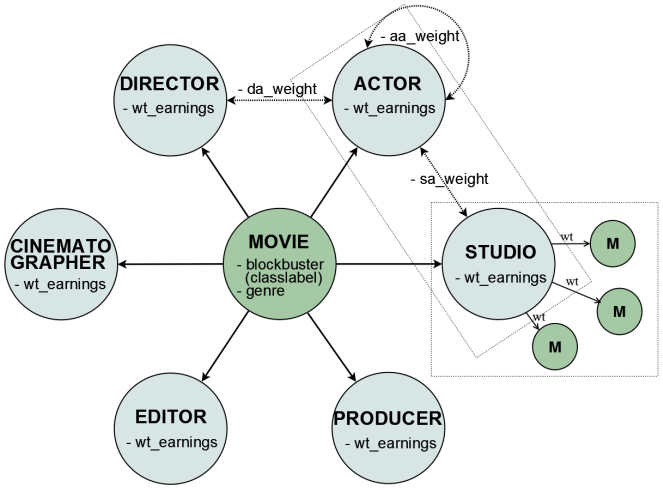


Figure 4.1. The relational schema for the IMDb dataset. The link weights 'da_weight', 'aa_weight' and 'sa_weight' are computed during graph summarization. The attribute 'wt_earnings' on Studios and other objects is computed by summarizing the gross earnings of the Movies(M) related to that object in the past.

In addition, it was necessary to preprocess the data to consolidate duplicate names. In particular, we focused on identifying the primary studios associated with each movie. We used pattern matching and manual filtering to merge duplicate studios while being careful not to merge similar (but not the same) entities as in Table 4.2. Further, we considered only the studios which produced at least ten movies since 1981.

The IMDb dataset is quite extensive but for the purpose of our experiments, we pruned it to conform to the schema shown in Figure 4.1 defined as a Qgraph schema [34]. The attributes associated with each object in Figure 4.1 are the attributes that were supplied to the relational classfication model. The classification task was to predict whether a movie would be a blockbuster (Gross earnings $> \$90mil$). The attribute 'wt_earnings' on all objects (except Movie) represents the weighted average earnings based on summarization of the movies of that object in past time steps—thus, there is a 'wt_earnings' attribute corresponding to each time step in TVRC. Further, the Actor-Actor, Actor-Director, Actor-Studio links and their respective

link weights namely 'aa_weight', 'da_weight' and 'sa_weight' are generated during the graph summarization phase. Note that we use the class label on related objects for prediction but only the labels corresponding to the previous time steps are available to the model (this also applies to the other two datasets).

4.1.2 Cora

Cora is a database of computer science research papers with the respective citation and author information. The dataset is available at *http://www.cs.umass.edu/∼mccallum/code-data.html*. The relational schema is given in Figure 4.2. The attributes associated with each object are those supplied to the relational classification model while the summary weights on the Paper-Citation and Author-Author links, namely 'pc_weight' and 'aa_weight', are generated by the graph summarization phase. The prediction task here was to predict the whether a paper is a machine learning paper given the topic of its references and the most prevalent topics its authors are working on through collaborations with other authors.

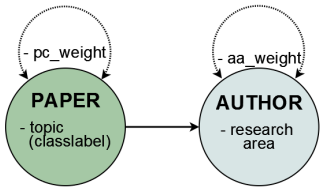


Figure 4.2. The relational schema for the Cora dataset. The link weights 'pc_weight' and 'aa_weight' are computed during graph summarization. The classlabel 'topic' is a binary classlabel showing whether the paper is a machine learning paper or not.

4.1.3 Reality Mining

The Reality Mining dataset captures communication, proximity, location, and activity information from 97 subjects (students, researchers, faculty, etc.) at MIT

over the course of the 2004-2005 academic year. Each participant of the study was equipped with a bluetooth Nokia cellphone. The data records the duration of phone calls between pairs of subjects as well as the extent of periods in which pairs of individuals were in proximity of one another. A more detailed description of the dataset is available at *http://www.reality.mit.edu*.

The database was pruned to conform to the relational schema in Figure 4.3 for our experiments. The prediction task was to predict whether a person is a student or not based on his/her call patterns and device proximity measurements. This dataset is different from the other two datasets for two reasons. Firstly, the number of objects in this dataset is quite small but the number of links between those objects is large. Secondly, in the earlier datasets, a new temporal event like publishing of a paper or a movie release is defined on a yearly basis, while in this dataset users are entering the system more quickly. Therefore, we had to reduce the granularity of our temporal dimension to a month instead of a year in this case.

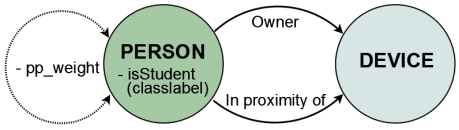


Figure 4.3. The relational schema for the Reality Mining dataset. The link weight 'pp_weight' is computed during graph summarization and captures the influence of both call relationships and device proximity relationships between two people.

4.2 Models

We evaluated three different models in a classification context to assess the performance of TVRC against baseline approaches:

- **Snapshot Model**: The Snapshot model is a baseline model that uses a graph $G_{\leq t}$, which consists of all objects and links up to and including the year t of

the sample, for learning. The Snapshot model does not weight the attributes or the links.

- **Window Model**: The Window model is again a baseline model that uses the graph G_{t-1} for prediction on graph G_t. In other words, it only uses the immediate past information for prediction and ignores all the other historical data. With respect to the kernel smoothing phase in TVRC, the Window model is identical to TVRC where the summary parameter θ=1.0.

- **TVRC**: As motivated earlier, we present the results of TVRC algorithm using an exponential smoothing kernel for summarization and Weighted Relational Classifier for prediction. The TVRC model first summarizes the graph into G_t^S using all years up to and including t, selecting the weighing parameter θ using k-fold cross validation and then weights the attributes appropriately during learning and prediction.

We describe the results of TVRC on three datasets (Cora, IMDb, Reality Mining) evaluating three different kernels (exponential, linear, inverse linear) for summarization and two relational classification models (RBC and RPT).

4.3 Methodology

For each dataset, we divided the data into disjoint temporal samples or 'snapshots' $\{G_1, G_2, \cdots, G_t\}$ where each snapshot G_i corresponds to the events that happened at time i. As discussed in Section 4.1.3, for the IMDb and Cora datasets, new events like a new movie release and publishing of a paper respectively are considered yearly. Therefore, for these two datasets our temporal dimension is defined in years. For IMDb, $t \in$ [2002,2007] while for Cora, $t \in$ [1993,1998]. For Reality Mining however, $t \in$ {July, August, September, October, November} due to the events (new people joining the system) having finer temporal granularity in this dataset. Note that the

summarization phase uses data from a larger time window for each dataset as given by Table 4.1. Table 4.3 summarizes the classification task on each of the datasets.

Table 4.3
Classification tasks.

Dataset	**Prediction task**
IMDb	Movie *blockbuster* status (gross> \$90*mil*)
Cora	Paper *topic* (is ML paper or not)
Reality Mining	*Student* vs. Faculty/Other

The classification experiments are set up as follows: we learn the model on the sample corresponding to time t and apply the model on the subsequent sample corresponding to time $(t + 1)$. The samples will vary depending on the model being evaluated. For example, while evaluating the TVRC model, we would learn the model on G_t^S and apply the model on G_{t+1}^S. However, for evaluating the Snapshot model, we would learn the model on $G_{\leq t}$ and apply the model on G_{t+1}. We compare the performance of different methods using area under the ROC curve (AUC).

4.4 Results and Analysis

Our first set of experiments assess the cross-validation aspect of TVRC. We compare TVRC to a ceiling model which chooses the optimal value for the summary parameter θ by evaluating the model for all values of θ on the test set and choosing the best value. Figure 4.4 shows the average AUC perfromance for different values of θ. The plot is an inverted curve with a unique maxima for all the three datasets—this indicates that there exists an optimal amalgamation point for historical and current information for the best prediction accuracies and using just the past information or the present data may not yield the best results. '$\square$' shows the optimal choice of θ for TVRC (θ_O) while '$\diamond$' shows the θ picked using k-fold cross validation (θ_{CV}). It is clear that choosing the summary parameter θ through cross-validation is not

significantly different from the optimal (ceiling) choice of θ (see paired t-test values in Table 4.4). This is notable because we use i.i.d. cross-validation here instead of relational cross-validation. Although it has been shown that ignoring dependencies among instances in relational domains may result in statistical biases [5], we conjecture that i.i.d. cross-validation works in this situation because all choices of parameter values are biased uniformly and thus, it does not affect the optimal parameter choice adversely.

Our second choice of experiments evaluate the performance of different relational classifiers on different datasets. Figure 4.5 shows the performance of each of the four models on the Cora classification task using the RBC and the RPT respectively as the relational classification models. We also compared the three models with ablated data to assess the temporal content in each type of link. Figures [4.5(c),4.5(d)] and Figures [4.5(e),4.5(f)] compare the performance of the three models when only the coauthor links and attributes or reference links and attributes are considered respectively. We differentiate between these two types of links as follows. Reference links occur only once in the snapshot G_t where t is the time when the paper was published. Thus, reference links are examples of *temporal events*. However, links between co-

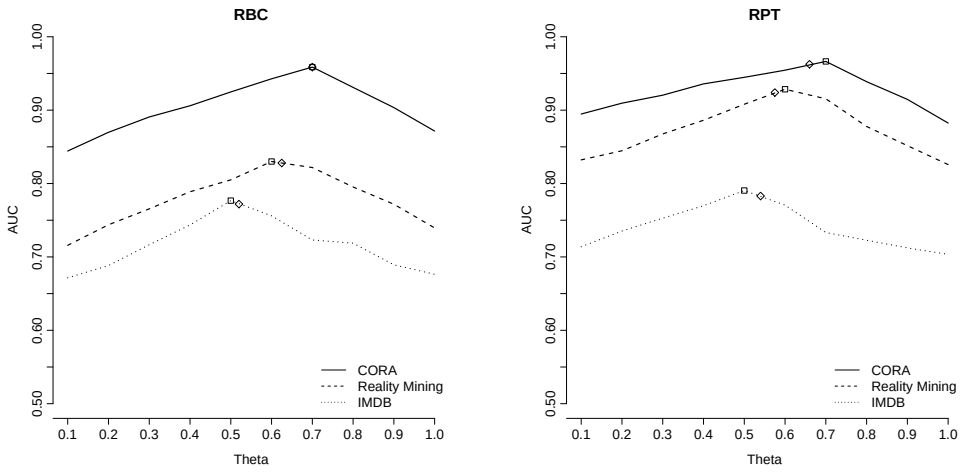
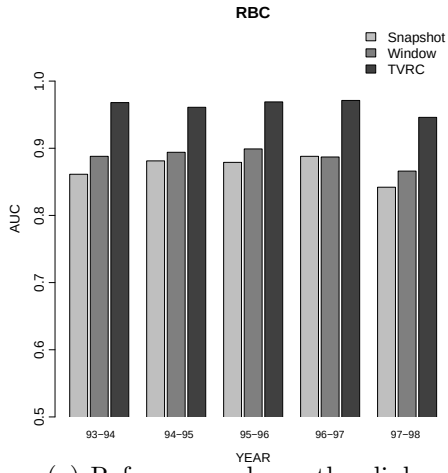
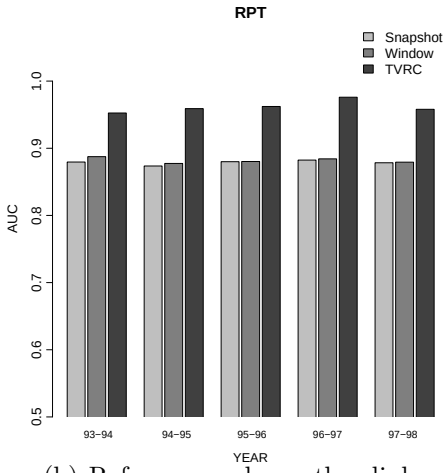


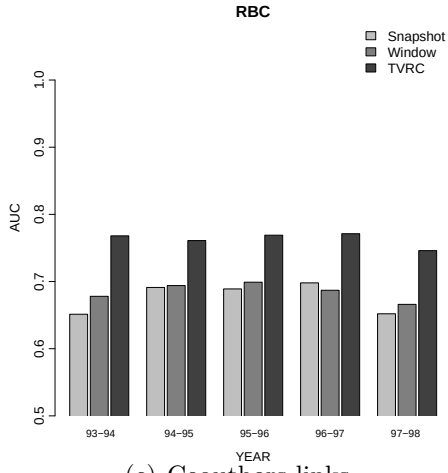
Figure 4.4. Comparing the values of parameter θ chosen through cross-validation '$\diamond$' θ_{CV} with the optimal value '$\square$' θ_O.



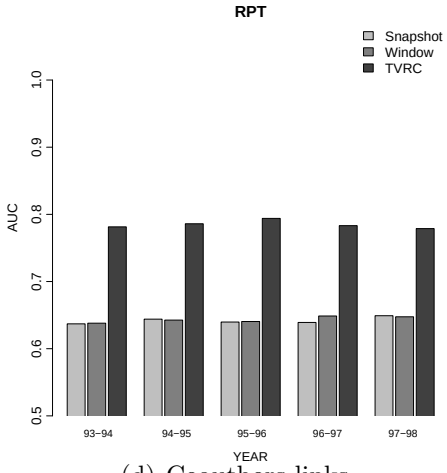
(a) Reference and coauthor links



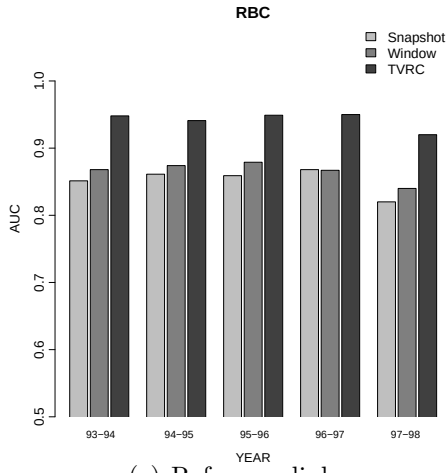
(b) Reference and coauthor links



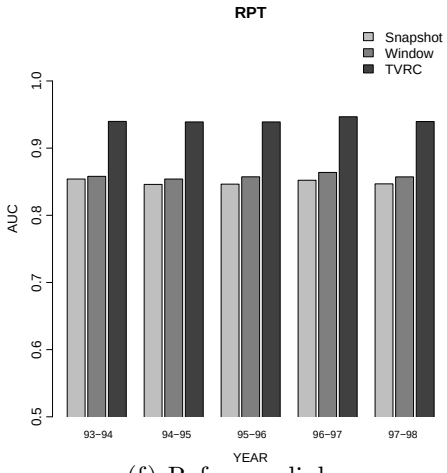
(c) Coauthors links



(d) Coauthors links



(e) Reference links



(f) Reference links

Figure 4.5. AUC performance on Cora dataset—TVRC uses exponential kernel for summarization.

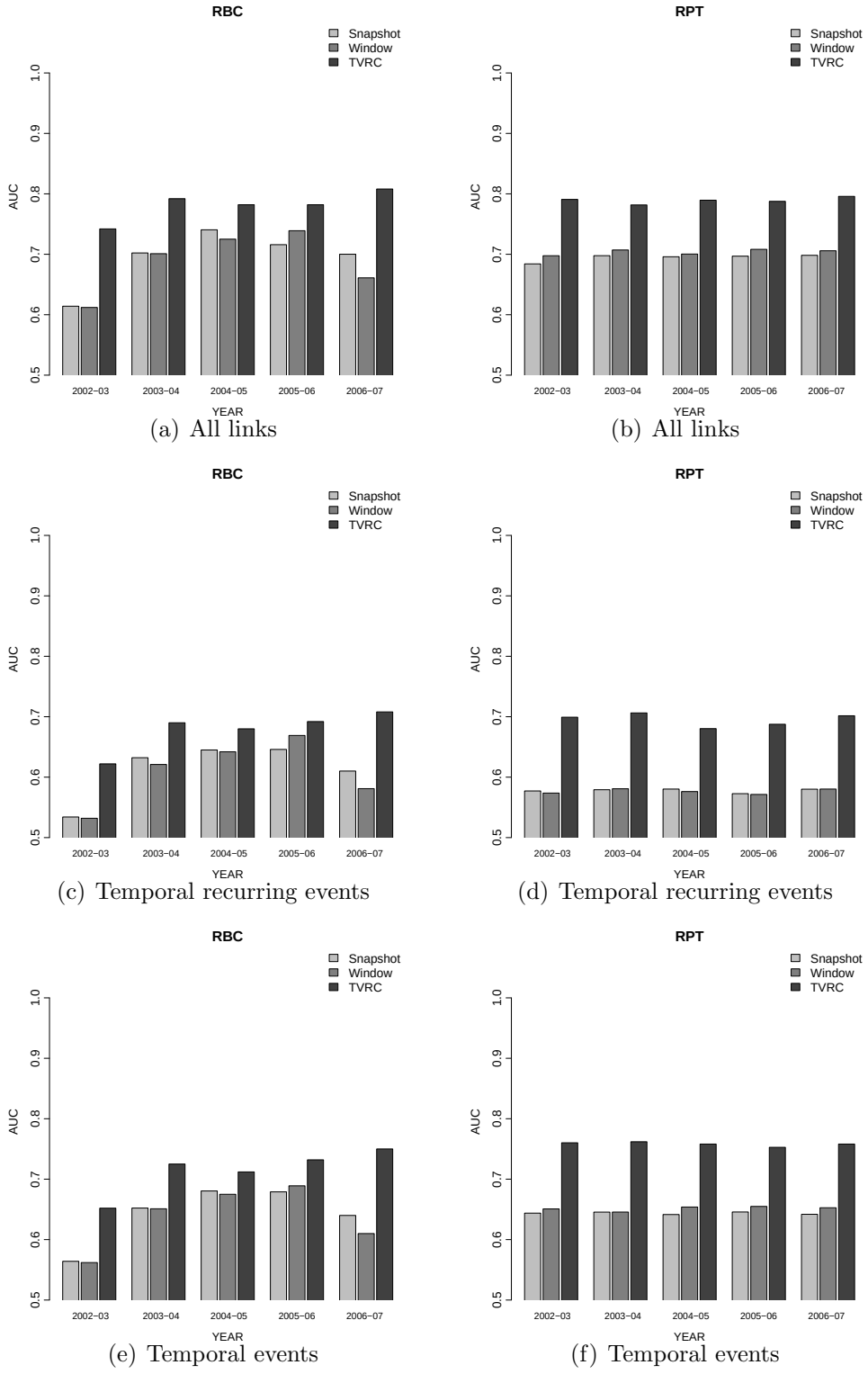
authors are examples of *temporally recurring events* as they recur everytime a new paper is published. Figure 4.5 clearly shows that TVRC results in a 10-15% improvement over both the baseline models regardless of the relational classifier used for prediction. In all three cases, the TVRC performs significantly better than baseline models. Furthermore, we also confirm that performance is more accurate using both the reference and coauthor attributes than either alone. In other words, both *temporal events* and *temporally recurring events* are useful in improving the performance of the model.

Figure 4.6 compares the performance of each of the three models on the IMDb classification task. The results show an improvement of 10-20% for TVRC over the baseline models for both the relational classifiers. Similar improvements are observed for ablated data experiments in Figures [4.6(c),4.6(d)] and Figures [4.6(e),4.6(f)] when the attributes corresponding to only the *temporal recurring events* (co-star) and *temporal events* (movies) are considered respectively.

Figure 4.7 compares the performance of each of the three models on the Reality Mining classification task. Again the results show an improvement of 10-15% for TVRC over the baseline models for both the classifiers. There were no *temporal events* in the Reality Mining in the relational schema considered. There are however several *temporally recurring events* as shown in Figure 4.3 namely the Person-Person call links and Person-Person device proximity links.

Our third set of experiments compares the performance of the different kernel functions used in the graph summarization phase for both relational models (RBC and RPT). Figure 4.8 compares the performance of the three kernels—exponential, linear and inverse linear—on each of the three input datasets. The results show that while all kernels improve prediction accuracy over baseline models, the exponential kernel is better than the other two kernels by at least 5% regardless of the model used for prediction.

To test the significance of the improvements in the AUC values, we provide the results of a two-tailed, paired t-test that compares the TVRC to each of the other three



(a) All links (b) All links

(c) Temporal recurring events (d) Temporal recurring events

(e) Temporal events (f) Temporal events

Figure 4.6. AUC performance on IMDb dataset—TVRC uses exponential kernel for summarization.

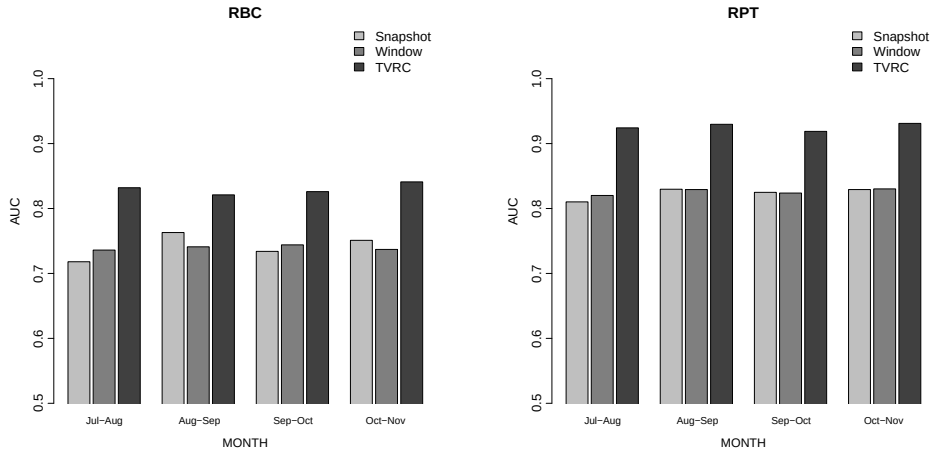


Figure 4.7. AUC performance on Reality Mining dataset—TVRC uses exponential kernel for summarization.

models across the set of trials for each dataset. Table 4.4 lists the p-values for each of these tests for TVRC using both the RBC and the RPT as the relational classifier. The improvement of the TVRC compared to each baseline models is significant at a $p < 0.01$ level, while the difference between the TVRC and the TVRC-Ceiling is not significant.

Table 4.4
Significance of TVRC improvement (paired t-test p-values). Here **S** refers to the Snapshot model, **W** refers to the Window model, **C** refers to TVRC-Ceiling.

Dataset	**S**	**W**	**C**	**S**	**W**	**C**
	(RBC)			(RPT)		
CORA	$1.0E^{-5}$	$5.0E^{-5}$	0.22	$1.7E^{-5}$	$3.0E^{-5}$	0.18
IMDb	$7.9E^{-4}$	$5.9E^{-4}$	0.18	$1.5E^{-5}$	$1.8E^{-5}$	0.26
Reality Mining	$5.5E^{-4}$	$4.4E^{-3}$	0.39	$1.5E^{-4}$	$1.5E^{-5}$	0.39

To test the significance of the improved performance using exponential kernel over linear and inverse linear kernels, we provide the results of a two-tailed, paired t-test

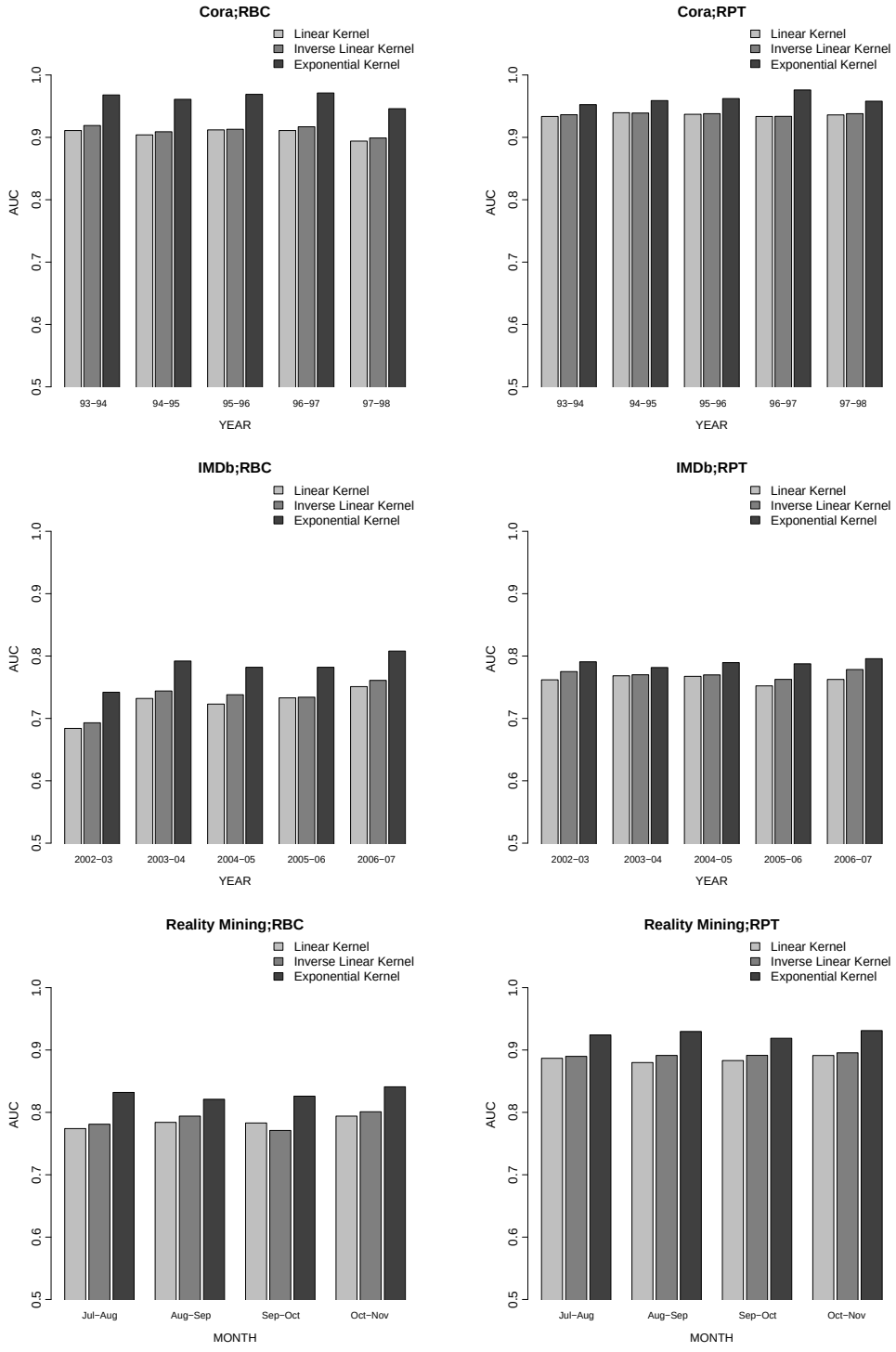


Figure 4.8. AUC performance of different kernels on the three datasets using both RBC and RPT for relational classification.

Table 4.5
Significance of exponential kernel performance compared to linear and inverse linear kernels (paired t-test p-values) for TVRC using RBCs.

Dataset	**Linear**	**Inverse Linear**
CORA	$1.9E^{-4}$	$2.8E^{-4}$
IMDb	$2.9E^{-6}$	$1.7E^{-4}$
Reality Mining	$1.9E^{-3}$	$6.3E^{-3}$

Table 4.6
Significance of exponential kernel performance compared to linear and inverse linear kernels (paired t-test p-values) for TVRC using RPTs.

Dataset	**Linear**	**Inverse Linear**
CORA	$1.7E^{-3}$	$1.7E^{-3}$
IMDb	$6.3E^{-3}$	$9.7E^{-3}$
Reality Mining	$5.8E^{-4}$	$3.3E^{-5}$

that compares the exponential kernel to the linear and inverse linear kernels across the set of trials for each dataset. Tables 4.5 and 4.6 list the p-values for each of these tests. The improvement of the exponential kernel compared to the other two kernels is significant at a $p < 0.01$ level.

Figure 4.9 compares the performance of the TVRC that uses the RBC as its component model (TVRC_{RBC}) to the TVRC using the RPT as its component model (TVRC_{RPT}) when exponential kernel is used in graph summarization phase. TVRC_{RPT} does almost as well as TVRC_{RBC} on CORA and IMDb - however, TVRC_{RPT} significantly outperforms TVRC_{RBC} on Reality Mining dataset due to a prevalence of useful degree and count features, which cannot be easily represented in a RBC. Table 4.7 lists the paired t-test results assessing the significance of performance difference TVRC_{RBC} and TVRC_{RPT}. The improvement of using the RPT over the RBC is significant at a

$p < 0.01$ level on the Reality Mining dataset while the difference on Cora and IMDb is not significant.

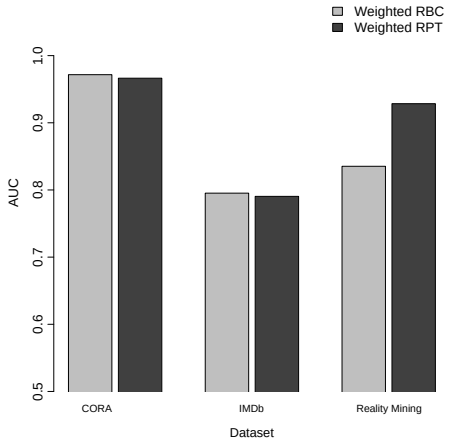


Figure 4.9. Comparing TVRC_{RBC} vs TVRC_{RPT}.

Table 4.7
Significance levels for TVRC_{RBC} vs TVRC_{RPT} (paired t-test p-values).

Dataset	**RBC vs RPT**
CORA	0.13
IMDb	0.53
Reality Mining	$2.0E^{-4}$

Thus, TVRC results in at least 10% improvement over both the baseline models—Snapshot model and Window model—for all the three datasets regardless of the prediction model used for relational classification. The improvements are also visible for any choice of kernel function employed—though in our experiments, the exponential kernel gives the best performance. The strength of TVRC lies in the modularity of the framework whereby each phase is independent of the other enabling us to choose the appropriate kernel function and relational classifier to best match the domain under consideration.

5 CONCLUSIONS

This thesis presents a new approach to modeling relational data with time-varying link structure. To date, work on statistical relational models has focused primarily on static snapshots of relational datasets even though most relational domains have temporal dynamics that are important to model. Although there has been some work modeling domains with time-varying attributes, to our knowledge this is the first model that exploits information in dynamic relationships between entities to improve prediction. This work has demonstrated that significant performance gains can be achieved by incorporating temporal-relational information into statistical relational models, even in a simple weighted summarization approach. We evaluated the algorithm on three real world domains and showed that our TVRC approach improves significantly upon the baseline approaches that ignore the temporal component of the data regardless of the kernel function used for summarization and relational model used for classification.

We evaluated the TVRC in three different domains, but the framework is applicable to a wide array of relational domains where the relationships between entities change over time. For example, our algorithm can be easily applied to predict the content category of a news magazine based on the references it makes. Also, our method of computing a weighted-summary graph can be combined with any statistical relational learner. We have chosen to use RBC and RPT in this work due to their simplicity. However, the TVRC framework is flexible and modular enough so as to use many different kernel functions for summarization and statistical relational models for classification (e.g., [7, 26, 35]).

One of the strengths of our approach is that it is a relatively simple and efficient way of incorporating time into statistical relational models. While our algorithm doesn't make a Markov assumption about the temporal dependencies, it is predicated

on the assumption that events in the recent past are more informative than events in the distant past. It also assumes that all relationships are meaningful (i.e., all edges are included in the summary graph calculation). A full joint *temporal-relational* model may be able to represent the dependencies in the data more accurately (e.g., [9]), however without a means to limit either the temporal or relational dependencies, the dimensionality of the model is far too large for accurate estimation with finite datasets.

This work attempts to model temporal dependencies by specifying a limited number of temporal patterns to moderate the relational dependencies. Additional efforts to identify and exploit temporal *motifs* for use as relational features (e.g., [29]) may be a promising means to extend the relational model space in a restricted way while still capturing most of the relevant temporal information in an efficient manner.

Possible future directions for this work include extending the temporal summarization schemes to model temporally varying attributes along with the link structure. In addition, we will consider formulating the TVRC framework as a latent variable model where the summary "influence" weights between pairs of nodes are hidden variables that change over time and affect the statistical dependencies between attribute values of incident nodes.

LIST OF REFERENCES

LIST OF REFERENCES

[1] S. Chakrabarti, B. Dom, and P. Indyk. Enhanced hypertext categorization using hyperlinks. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pages 307–318, 1998.

[2] P. Domingos and M. Richardson. Mining the network value of customers. In *Proceedings of the 7th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 57–66, 2001.

[3] J. Neville, O. Şimşek, D. Jensen, J. Komoroske, K. Palmer, and H. Goldberg. Using relational knowledge discovery to prevent securities fraud. In *Proceedings of the 11th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 449–458, 2005.

[4] M. McPherson, L. Smith-Lovin, and J. Cook. Birds of a feather: Homophily in social networks. *Annual Review of Sociology*, 27:415–445, 2001.

[5] D. Jensen and J. Neville. Linkage and autocorrelation cause feature selection bias in relational learning. In *Proceedings of the 19th International Conference on Machine Learning*, pages 259–266, 2002.

[6] C. Cortes, D. Pregibon, and C. Volinsky. Communities of interest. In *Proceedings of the 4th International Symposium of Intelligent Data Analysis*, pages 105–114, 2001.

[7] B. Taskar, P. Abbeel, and D. Koller. Discriminative probabilistic models for relational data. In *Proceedings of the 18th Conference on Uncertainty in Artificial Intelligence*, pages 485–492, 2002.

[8] L. Getoor and B. Taskar, editors. *Introduction to Statistical Relational Learning*. MIT Press, 2007.

[9] S. Sanghai, P. Domingos, and D. Weld. Learning statistical models of time-varying relational data. In *Proceedings of the Workshop on Statistical Relational Learning, 18th International Joint Conference on Artificial Intelligence*, 2003.

[10] D. Cai, Z. Shao, X. He, X. Yan, and J. Han. Community mining from multi-relational networks. In *Proceedings of the 9th European Conference on Principles and Practice of Knowledge Discovery in Databases*, 2005.

[11] S. Hill, D. Agarwal, R. Bell, and C. Volinsky. Building an effective representation of dynamic networks. *Journal of Computational and Graphical Statistics*, Sept, 2006.

[12] A. McCallum, K. Nigam, J. Rennie, and K. Seymore. A machine learning approach to building domain-specific search engines. In *Proceedings of the 16th International Joint Conference on Artificial Intelligence*, pages 662–667, 1999.

[13] Umang Sharan and Jennifer Neville. Exploiting time-varying relationships in statistical relational models. In *Proceedings of the Joint 9th WebKDD and 1st SNA-KDD on Web Mining and Social Network Analysis*, 2007.

[14] Umang Sharan and Jennifer Neville. A framework for exploiting temporal variations in relational domains. Under Submission, 2008.

[15] P. Kennedy. *A Guide to Econometrics*. The MIT Press, Cambridge, Massachusetts, 1998.

[16] H. Tanizaki. Bias correction of OLSE in the regression model with lagged dependent variables. *Computational Statistics and Data Analysis*, 34:495–511, 2000.

[17] R. Engle, editor. *ARCH: Selected Readings*. Oxford University Press, Oxford, 1995.

[18] J. Wooldrige. *Introductory Econometrics: A Modern Approach*. South-Western College Pub, 2003.

[19] T. Dean and K. Kanazawa. A model for reasoning about persistence and causation. *Computational Intelligence*, pages 142–150, 1989.

[20] Z. Ghahramani. Learning dynamic Bayesian networks. In *Adaptive Processing of Sequences and Data Structures*, pages 168–197. Springer-Verlag, 1998.

[21] C. Cortes, D. Pregibon, and C. Volinsky. Computational methods for dynamic graphs. *Journal of Computational and Graphical Statistics*, 2003.

[22] S. Hill, F. Provost, and C. Volinsky. Network-based marketing: Identifying likely adopters via consumer networks. *Statistical Science*, 22(2), 2006.

[23] Steve Hanneke and Eric Xing. Discrete temporal models of social networks. In *Proceedings of the ICML Workshop on Statistical Network Analysis*, 2006.

[24] Fan Guo, Steve Hanneke, Wenjie Fu, and Eric Xing. Recovering temporally rewiring networks: A model-based approach. In *Proceedings of the 24th International Conference on Machine Learning (ICML)*, 2007.

[25] T. Snijders. Markov chain monte carlo estimation of exponential random graph models. *Journal of Social Structure*, 3(2), 2002.

[26] N. Friedman, L. Getoor, D. Koller, and A. Pfeffer. Learning probabilistic relational models. In *Proceedings of the 16th International Joint Conference on Artificial Intelligence*, pages 1300–1309, 1999.

[27] L. Getoor, N. Friedman, D. Koller, and A. Pfeffer. Learning probabilistic relational models. In *Relational Data Mining*, pages 307–335. Springer-Verlag, 2001.

[28] L. Getoor, N. Friedman, D. Koller, and B. Taskar. Learning probabilistic models with link uncertainty. *Journal of Machine Learning Research*, 3:679–707, 2002.

[29] Amy McGovern, Nathan C. Hiers, Matthew Collier, and Rodger A. Brown. Spatiotemporal relational probability trees. In *Submission to the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2008.

[30] J. Neville, D. Jensen, L. Friedland, and M. Hay. Learning relational probability trees. In *Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 625–630, 2003.

[31] B. Milch, B. Marthi, S. Russell, D. Sontag, D. Ong, and A. Kolobov. Blog: Probabilistic models with unknown objects. In *Introduction to Statistical Relational Learning*, 2007.

[32] J. Neville, D. Jensen, and B. Gallagher. Simple estimators for relational Bayesian classifers. In *Proceedings of the 3rd IEEE International Conference on Data Mining*, pages 609–612, 2003.

[33] P. Domingos and M. Pazzani. On the optimality of the simple Bayesian classifier under zero-one loss. *Machine Learning*, 29:103–130, 1997.

[34] H. Blau, N. Immerman, and D. Jensen. A visual query language for relational knowledge discovery. Technical Report 01-28, University of Massachusetts Amherst, Computer Science Department, 2001.

[35] C. Perlich and F. Provost. Aggregation-based feature invention and relational concept classes. In *Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 167–176, 2003.

APPENDIX

A FULL EXPERIMENTAL RESULTS

Table A.1: AUC values for classification using Weighted RBC summarized using the exponential kernel. SM denotes the Snapshot model, W stands for the Window model, the summary parameter $\theta \in [0.1,0.9]$. θ_O is the optimal parameter computed by TVRC (Ceiling) while θ_{CV} is the parameter selected by TVRC through cross validation. $CORA_{tr}$ is the dataset where only temporally recurring events are considered (in this case coauthorship relationships), $CORA_{te}$ is the dataset where only temporal events are considered (citation relationships) whereas $CORA_{all}$ considers all types of links. The same notation is followed for IMDB as well.

Time	**SM**	**0.1**	**0.2**	**0.3**	**0.4**	**0.5**	**0.6**	**0.7**	**0.8**	**0.9**	**WM**	θ_O	θ_{CV}
$CORA_{all}$													
1993-94	0.861	0.868	0.894	0.922	0.924	0.935	0.947	0.968	0.951	0.931	0.888	0.7	0.7
1994-95	0.881	0.874	0.893	0.916	0.930	0.944	0.961	0.973	0.944	0.923	0.894	0.7	0.6
1995-96	0.879	0.847	0.898	0.910	0.921	0.933	0.954	0.969	0.945	0.929	0.899	0.7	0.7
1996-97	0.888	0.889	0.897	0.913	0.906	0.947	0.959	0.971	0.941	0.913	0.887	0.7	0.7
1997-98	0.842	0.838	0.868	0.889	0.926	0.939	0.963	0.977	0.946	0.902	0.866	0.7	0.8
$CORA_{tr}$													
1993-94	0.651	0.665	0.687	0.721	0.725	0.738	0.746	0.768	0.750	0.731	0.678	0.7	0.7
1994-95	0.691	0.674	0.693	0.715	0.735	0.745	0.761	0.774	0.744	0.723	0.694	0.7	0.6
1995-96	0.689	0.647	0.698	0.711	0.721	0.733	0.754	0.769	0.745	0.729	0.699	0.7	0.7
1996-97	0.698	0.689	0.697	0.718	0.727	0.745	0.759	0.771	0.741	0.713	0.687	0.7	0.7
1997-98	0.652	0.638	0.668	0.689	0.726	0.741	0.763	0.777	0.746	0.702	0.666	0.7	0.8
$CORA_{te}$													
1993-94	0.851	0.868	0.884	0.903	0.917	0.929	0.937	0.948	0.931	0.909	0.868	0.7	0.7
1994-95	0.861	0.874	0.883	0.906	0.932	0.944	0.941	0.953	0.924	0.902	0.873	0.7	0.6
1995-96	0.859	0.847	0.876	0.891	0.903	0.916	0.937	0.949	0.928	0.911	0.882	0.7	0.7
1996-97	0.868	0.869	0.878	0.894	0.906	0.927	0.938	0.952	0.924	0.892	0.864	0.7	0.7
1997-98	0.822	0.838	0.856	0.871	0.897	0.916	0.923	0.934	0.921	0.902	0.843	0.7	0.8
$IMDB_{all}$													
2002-03	0.614	0.610	0.633	0.674	0.715	0.742	0.713	0.685	0.701	0.625	0.612	0.5	0.5
2003-04	0.702	0.676	0.718	0.747	0.765	0.792	0.775	0.739	0.742	0.722	0.701	0.5	0.5
2004-05	0.741	0.739	0.733	0.763	0.775	0.782	0.777	0.764	0.739	0.738	0.725	0.5	0.5
2005-06	0.716	0.718	0.731	0.753	0.771	0.799	0.782	0.768	0.754	0.741	0.739	0.5	0.6
2006-07	0.700	0.662	0.675	0.704	0.722	0.808	0.765	0.726	0.715	0.671	0.661	0.5	0.5
$IMDB_{tr}$													
2002-03	0.534	0.552	0.573	0.594	0.615	0.622	0.613	0.595	0.571	0.558	0.532	0.5	0.5
2003-04	0.632	0.646	0.658	0.674	0.685	0.692	0.675	0.649	0.632	0.625	0.621	0.5	0.5
2004-05	0.625	0.631	0.643	0.651	0.667	0.682	0.671	0.654	0.646	0.633	0.622	0.5	0.5
2005-06	0.626	0.642	0.653	0.668	0.675	0.692	0.687	0.683	0.668	0.643	0.629	0.5	0.6
2006-07	0.611	0.622	0.639	0.664	0.682	0.708	0.685	0.667	0.645	0.621	0.591	0.5	0.5
$IMDB_{te}$													
2002-03	0.564	0.582	0.603	0.614	0.635	0.652	0.643	0.625	0.591	0.579	0.562	0.5	0.5
2003-04	0.632	0.643	0.664	0.689	0.704	0.725	0.709	0.686	0.665	0.652	0.631	0.5	0.5
2004-05	0.661	0.679	0.685	0.693	0.704	0.712	0.707	0.693	0.679	0.668	0.665	0.5	0.5
2005-06	0.659	0.678	0.691	0.713	0.729	0.741	0.732	0.718	0.694	0.671	0.662	0.5	0.6
2006-07	0.642	0.662	0.675	0.704	0.722	0.753	0.735	0.706	0.682	0.651	0.634	0.5	0.5
$REAL$													
Jul-Aug	0.718	0.695	0.713	0.737	0.788	0.811	0.832	0.825	0.798	0.773	0.736	0.6	0.6
Aug-Sep	0.763	0.738	0.766	0.783	0.794	0.802	0.821	0.817	0.794	0.771	0.741	0.6	0.6

Continued on next page ...

Table A.1 – continued from previous page

Time	SM	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	WM	θ_O	θ_{CV}
Sep-Oct	0.734	0.701	0.722	0.761	0.781	0.810	0.826	0.847	0.793	0.769	0.744	0.7	0.6
Oct-Nov	0.751	0.729	0.774	0.781	0.792	0.797	0.841	0.838	0.796	0.774	0.737	0.6	0.6

Table A.2: AUC values for classification using Weighted RPT. The same notation is followed as described in Table A.1

Time	SM	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	WM	θ_O	θ_{CV}
$CORA_{all}$													
1993-94	0.879	0.892	0.913	0.925	0.932	0.944	0.953	0.962	0.935	0.914	0.885	0.7	0.6
1994-95	0.874	0.896	0.911	0.921	0.936	0.940	0.952	0.959	0.934	0.916	0.877	0.7	0.6
1995-96	0.881	0.897	0.909	0.919	0.928	0.941	0.959	0.962	0.939	0.912	0.883	0.7	0.7
1996-97	0.882	0.906	0.918	0.927	0.937	0.948	0.957	0.976	0.937	0.915	0.884	0.7	0.7
1997-98	0.878	0.886	0.905	0.916	0.936	0.947	0.958	0.971	0.949	0.916	0.879	0.7	0.8
$CORA_{tr}$													
1993-94	0.637	0.652	0.685	0.706	0.726	0.749	0.781	0.804	0.750	0.694	0.638	0.7	0.7
1994-95	0.644	0.658	0.686	0.697	0.726	0.751	0.779	0.786	0.751	0.684	0.643	0.7	0.6
1995-96	0.639	0.652	0.686	0.707	0.721	0.752	0.775	0.794	0.752	0.706	0.641	0.7	0.7
1996-97	0.639	0.658	0.682	0.706	0.719	0.751	0.776	0.783	0.748	0.705	0.648	0.7	0.7
1997-98	0.649	0.662	0.687	0.706	0.725	0.751	0.779	0.801	0.747	0.705	0.641	0.7	0.8
$CORA_{te}$													
1993-94	0.854	0.875	0.891	0.913	0.922	0.936	0.939	0.943	0.903	0.876	0.858	0.7	0.7
1994-95	0.846	0.873	0.896	0.916	0.926	0.931	0.936	0.939	0.902	0.873	0.854	0.7	0.6
1995-96	0.846	0.876	0.897	0.916	0.929	0.936	0.939	0.942	0.906	0.876	0.857	0.7	0.7
1996-97	0.852	0.877	0.896	0.917	0.927	0.937	0.940	0.947	0.907	0.878	0.864	0.7	0.7
1997-98	0.847	0.873	0.896	0.916	0.926	0.936	0.941	0.948	0.905	0.876	0.857	0.7	0.8
$IMDB_{all}$													
2002-03	0.684	0.719	0.737	0.754	0.769	0.791	0.767	0.732	0.725	0.717	0.698	0.5	0.5
2003-04	0.698	0.709	0.734	0.756	0.766	0.782	0.763	0.735	0.718	0.708	0.707	0.5	0.5
2004-05	0.696	0.708	0.731	0.755	0.779	0.793	0.789	0.728	0.728	0.714	0.703	0.5	0.5
2005-06	0.697	0.715	0.739	0.749	0.767	0.799	0.788	0.734	0.719	0.711	0.708	0.5	0.6
2006-07	0.698	0.714	0.731	0.748	0.769	0.796	0.761	0.733	0.728	0.715	0.706	0.5	0.5
$IMDB_{tr}$													
2002-03	0.577	0.596	0.614	0.644	0.675	0.699	0.665	0.635	0.614	0.592	0.574	0.5	0.5
2003-04	0.579	0.594	0.616	0.640	0.674	0.706	0.667	0.637	0.608	0.587	0.581	0.5	0.5
2004-05	0.580	0.601	0.611	0.643	0.673	0.694	0.680	0.637	0.617	0.595	0.576	0.5	0.5
2005-06	0.573	0.598	0.617	0.642	0.675	0.702	0.688	0.638	0.617	0.595	0.571	0.5	0.6
2006-07	0.580	0.601	0.621	0.642	0.671	0.706	0.663	0.640	0.614	0.597	0.580	0.5	0.5
$IMDB_{te}$													
2002-03	0.644	0.668	0.688	0.719	0.732	0.760	0.741	0.712	0.709	0.683	0.651	0.5	0.5
2003-04	0.646	0.678	0.692	0.718	0.738	0.762	0.737	0.722	0.707	0.682	0.646	0.5	0.5
2004-05	0.642	0.673	0.688	0.713	0.735	0.761	0.758	0.711	0.693	0.681	0.654	0.5	0.5
2005-06	0.646	0.668	0.685	0.709	0.734	0.762	0.753	0.714	0.704	0.675	0.655	0.5	0.6
2006-07	0.642	0.667	0.688	0.713	0.742	0.758	0.737	0.713	0.695	0.682	0.653	0.5	0.5
$REAL$													
Jul-Aug	0.810	0.823	0.837	0.861	0.882	0.901	0.924	0.919	0.877	0.854	0.820	0.6	0.6
Aug-Sep	0.829	0.835	0.845	0.869	0.887	0.902	0.929	0.918	0.879	0.852	0.829	0.6	0.6
Sep-Oct	0.825	0.833	0.848	0.871	0.887	0.919	0.928	0.908	0.871	0.850	0.824	0.7	0.6
Oct-Nov	0.829	0.838	0.849	0.869	0.889	0.910	0.931	0.917	0.884	0.851	0.832	0.6	0.6

Table A.3: AUC values for classification using linear and inverse linear kernels

	Weighted RBC		**Weighted RPT**	
Time	**Linear**	**Inverse Linear**	**Linear**	**Inverse Linear**
$CORA_{all}$				
1993-94	0.931	0.939	0.934	0.936
1994-95	0.934	0.939	0.939	0.939
1995-96	0.932	0.933	0.938	0.937
1996-97	0.941	0.947	0.934	0.934
1997-98	0.934	0.939	0.936	0.938
$CORA_{tr}$				
1993-94	0.741	0.738	0.769	0.771
1994-95	0.739	0.745	0.769	0.768
1995-96	0.742	0.748	0.762	0.769
1996-97	0.744	0.746	0.779	0.775
1997-98	0.745	0.756	0.762	0.762
$CORA_{te}$				
1993-94	0.914	0.913	0.913	0.916
1994-95	0.925	0.918	0.919	0.918
1995-96	0.926	0.917	0.917	0.918
1996-97	0.903	0.904	0.914	0.919
1997-98	0.923	0.919	0.918	0.922
$IMDB_{all}$				
2002-03	0.704	0.713	0.762	0.775
2003-04	0.762	0.774	0.768	0.771
2004-05	0.753	0.768	0.768	0.769
2005-06	0.763	0.764	0.752	0.763
2006-07	0.771	0.772	0.763	0.778
$IMDB_{tr}$				
2002-03	0.591	0.593	0.675	0.679
2003-04	0.662	0.664	0.664	0.674
2004-05	0.655	0.659	0.656	0.684
2005-06	0.655	0.656	0.669	0.673
2006-07	0.669	0.671	0.669	0.673
$IMDB_{te}$				
2002-03	0.626	0.629	0.728	0.736
2003-04	0.681	0.682	0.729	0.739
2004-05	0.675	0.678	0.723	0.733
2005-06	0.702	0.705	0.730	0.734
2006-07	0.711	0.721	0.729	0.733
$REAL$				
Jul-Aug	0.774	0.781	0.887	0.889
Aug-Sep	0.784	0.794	0.881	0.891
Sep-Oct	0.783	0.771	0.883	0.891
Oct-Nov	0.794	0.801	0.891	0.896