Second Order Pseudolikelihood Learning in Relational Domain

Krishna Kumar Tiwari, V. Vijaya Saradhi

Indian Institute of Technology Guwahati,
Guwahati, India.
Email: {k.tiwari@iitg.ernet.in, saradhi@iitg.ernet.in}

Abstract. We use composite likelihood for structure learning and parameter estimation in relational dependency networks (RDNs). RDNs currently use pseudolikelihood, to learn parameters, which is a special case of composite likelihood function. Composite likelihood learning is used to give trade-off between computational complexity and performance of the model. Variance of the model is minimum in case of full likelihood and maximum in pseudolikelihood.

In particular we focus on modified second order pseudolikelihood function and extend relational Bayesian classifier (RBC) to this setting. Second order RDNs explore pairwise attribute correlation. We evaluate second order learning on synthetic and real world data sets. We observe experimentally second order model has an edge over the pseudolikelihood based model particularly when correlation is high.

1 Introduction

In real world, data set entities are related to each other either directly or indirectly. As noted in [18], labels of linked pages are highly correlated in web page documents. In case of patent cross-citations, scientific papers share relationship based on common author, editor, publisher. In social networks connected people share similar behaviors. In traditional machine learning domain objects are homogeneous, data is stored in single table with each row representing a new instance. In real world data sets there are multiple tables to represent different objects and relationship between the objects. Traditional machine learning algorithms are employed by *flattening* the data. *Flattened* data loses rich relational information. Consider the case of citation database[1], if we use only attributes of paper to predict the topic, then we lose the relation information given by its referenced and cited papers.

In relational learning one challenge is how to model the data. Many graphical models are often used to represent the relational data. Graphical models may be directed like relational Bayesian networks [9] [10] or undirected like relational Markov networks [19], each having some advantages and disadvantages. Directed relational Bayesian networks does not allow cyclic dependencies among

[1] Collection of previous year papers, data base represent a network of cited papers.

Natarajan Meghanathan et al. (Eds) : DMAP, CSITA, ARIN, ISPR, AISO - 2019
pp. 21-33, 2019. © CS & IT-CSCP 2019 DOI: 10.5121/csit.2019.91603

the attributes but has a simple parameter estimation technique [9]. Undirected relational Markov network is able to represent cyclic dependencies but requires known network structure and parameter estimation requires repeated inference over large values [6].

Another crucial property of relational data is autocorrelation (Jensen and Neville, 2002), which refers to correlation between values of the same variable on related objects. It is a common characteristic of many datasets. Autocorrelation creates instance dependencies which violate the principle of traditional machine learning algorithms [14]. Autocorrelation provides an opportunity to apply collective inference to increase the performance [14].

2 Related Work

RDNs are the first model to represent autocorrelation [14]. RDN is an undirected model which uses simple parameter estimation technique similar to relational Bayesian networks. When compared with relational Bayesian networks and relational Markov networks, RDN is an approximate model. RDN calculates all conditional probability tables independently. The authors assure that collective inference can increase the performance compare to individual inference models [12]. RDN uses pseudolikelihood learning which is an approximation of maximum likelihood. RDN is not guaranteed to be consistent [2], because of the approximate nature. Gibbs sampling is used to apply inference to enhance the model performance and guarantees the consistency of the model by recovering full joint distribution [7]. Next, we will discuss different approximation methods of full likelihood.

Maximum likelihood is the most useful parameter estimation technique in machine learning but it follows the independent assumption of instances. Assume data contains n samples of m dimensional vectors following $i.i.d$, sampled from a distribution p_{θ_0} with $\theta_0 \in \Theta \subset R^r$, $D = (X^1, ...X^n)$, $\quad X^i \in R^m$. Maximum likelihood estimator (MLE) $\widehat{\theta}_n^{ml}$ is a maximization of log-likelihood function [4].

$$l_n(\theta; D) = \sum_{i=1}^{n} \log p_\theta(X^i) \tag{1}$$
$$\widehat{\theta}_n^{ml} = \underset{\theta_0 \, \in \, \Theta}{\arg\max}\, l_n(\theta; D)$$

Properties of MLE estimate :

- Consistent: $\lim_{n\to\infty} \widehat{\theta}_n^{ml} = \theta_0$, as the number of samples grows, estimator will converge to true parameter θ_0 [4].
- Smallest possible variance $(nI(\theta_0)^{-1})$ (Ferguson 1996).
- Computationally intensive due to normalization factor.

[2] consistency is used in the context of local CPDs ([13], [8])

In some situations like in high dimensionality of data and cyclic dependencies, use of MLE is intractable. In those situations approximation methods come into picture. Approximation methods treat the joint likelihood function as product of likelihood objects [1], these are called product pseudolikelihoods. Likelihood objects are formed by considering subset of variables. Assuming subscript represents dimension of that item and $X^i_{-j} = \{X^i_k \ : k \neq j\}$, pseudolikelihood (PL) [2] is defined as :

$$pl_n(\theta; D) = \sum_{i=1}^{n} \sum_{j=1}^{m} \log p_\theta(X^i_j | X^i_{-j}). \tag{2}$$

Properties of PL estimate:

- Consistent : $\lim_{n\to\infty} \widehat{\theta}^{ml}_n = \theta_0$ under regularity condition[3].
- Higher asymptotic variance.
- Low computational complexity because it does not require computation of normalization factor.

Composite likelihood is a generalization of pseudolikelihood function. Assume we have different order pseudolikelihood objects then we can combine them in a composite function called composite likelihood given by Lindsay in 1988. Likelihood object can be viewed as $S_\theta = logp_\theta(X_A|X_B)$, where A and B represents the dimension set of the instance X. According to definition of pseudolikelihood objects , $A \neq \phi = A \cap B$. Composite likelihood function is defined as:

$$cl_n = \sum_{i=1}^{n} \sum_{j=1}^{k} \log p_\theta(X^i_{A_j} | X^i_{B_j}). \tag{3}$$

Based on the cardinality of set A_j in (3), we get $PL(1), PL(2), ...(FL)$ (full likelihood). Dillon extended composite likelihood by introducing component weight and selection probabilities to make it stochastic composite likelihood [4].
Properties of SCL:

- Strongly consistent[4].
- Variance is between PL and FL.
- Computation cost varies based on the selection of object, minimum in special case of PL and maximum in case FL.

As mentioned in [1], different order of likelihood object lead us to estimator of different efficiencies.

We introduce higher order pseudolikelihood objective function in RDN learning. Our motivation is to reach close to the maximum likelihood in approximation. Attribute correlation can be easily modeled by using this technique. We

[3] Jennifer Neville proved the consistency of pseudolikelihood in regularity conditions. Details of standard regularity conditions are discussed in [13].

[4] Dillon proves the consistency of the estimator [3].

observe that in highly correlated data higher order learning would be a better idea. It can increase the performance as well as likelihood estimate.

In this paper we introduce second order pseudolikelihood learning in RDN. We explore the attribute pairs which are highly correlated. To our knowledge there is no individual learner which follows the second order PL technique, which motivates us to extend Bayesian classifier to this setting [5]. Second order Bayesian classifier uses same estimation technique used in case of relational Bayesian classifier [16]. We demonstrated gains in performance in highly correlated environment.

We discuss our approach in more detail below. Fist we discuss second order PL function in the context of RDN's in section 3. Next , we described the second order relational Bayesian classifier in section 4. Then we evaluate our approach on synthetic and real data sets in section 6. We conclude the work in section 7.

3 Second Order Pseudolikelihood

RDN uses three graphs to represent the whole model. Object and relations are represented by data graph (Figure 1), $G_D = (V_D, E_D)$. V_D represents objects in the data graph like paper, author etc. E_D represents relationship between these objects such as cites, author_of in case of citation network. RDN learns attribute relations by relational learners and represent them in model graph (Figure 2). In this figure, the paper object has identified relations among the attributes namely topic, year, type and month. RDN assigns soft class label by model graph and uses Gibbs sampling to apply collective inference on inference graph (Figure 3) to improve model performance. Each node and link is associated with a type $T(v_i) = t_{v_i}$ and $T(e_j) = t_{e_j}$. Objects are heterogeneous and have number of associated attributes $X^t = \{X_1^t, ...X_m^t\}$ based on its type $t \in T$. RDN represents a joint distribution over the values of the attributes in the data

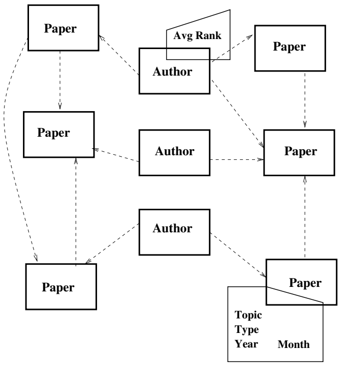


Fig. 1. data graph [14]

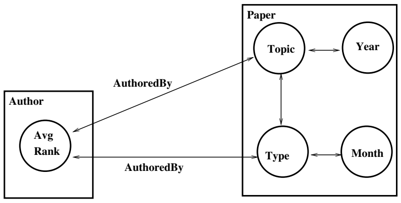


Fig. 2. model graph [14]

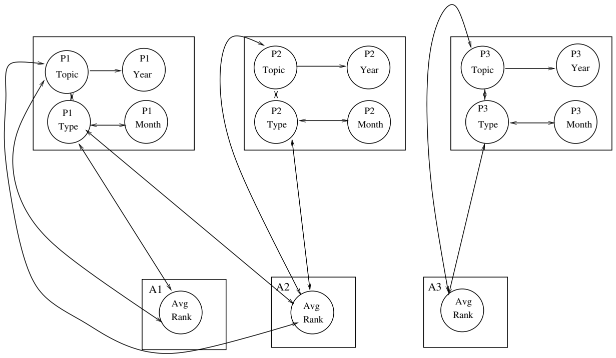


Fig. 3. inference graph [14]

graph,

$$x = \{\{X_{v_i}^{t_{v_i}} : v_i \in V \; s.t. \; T(v_i) = t_{v_i}\} \cup \{X_{e_j}^{t_{e_j}} : e_j \in E \; s.t. \; T(e_j) = t_{e_j}\}\}$$

Currently approximation of $p(x)$ is done by pseudolikelihood. RDN defines the pseudo-likelihood over item type, and is computed for data graph G_D as:

$$PL(G_D;\theta) = \prod_{t \in T} \prod_{X_i^t \in X^t} \prod_{v:T(v)=t} p(x_{v_i}^t | pa_{x_{v_i}^t}; \theta) \prod_{e:T(e)=t} p(x_{e_i}^t | pa_{x_{e_i}^t}; \theta) \qquad (4)$$

Where T represents set of item types, X^t denotes the set of attributes of type t. Take a closer look of formula when considering the variable $x_{v_i}^t$, it is conditioned on the values of its parents $pa_{x_{v_i}^t}$. Equation (4) assumes that calculation of any variable in its parents does not depend on $x_{v_i}^t$ (acyclicity condition similar to PRMs).

We observe that RDN can use composite likelihood function for structure learning and parameter estimation. Equation (5) computes composite likelihood of data graph G_D.

$$cl(G_D;\theta) = \prod_{t \in T} \prod_{X_{A_i}^t \in X^t}^{k} \prod_{v:T(v)=t} p(x_{v_{A_i}}^t | pa_{x_{v_{B_i}}^t}) \prod_{e:T(e)=t} p(x_{e_{A_i}}^t | pa_{x_{e_{B_i}}^t}) \qquad (5)$$

Subject to the constraint:

$$A_i \neq \emptyset = A_i \cap B_i$$

We get different estimators of varying efficiencies by choosing cardinality of set A_i. We deal with the case when cardinality of set A is 2, and we call this as second order pseudolikelihood. We employ second order PL instead of PL. Intuition for employing second order model stems from the arguments given in [1], [4]. More specifically, we observe two attributes together to explore the correlation between them. In equation (5) considering the variable $x^t_{v_{A_i}}$, we conditioned on the value $pa_{x^t_{v_{B_i}}}$, where $pa_{x^t_{vB_i}}$ is parent of $x^t_{v_{A_i}}$. $pa_{x^t_{v_{B_i}}}$ contains other attributes of same or related item by maintaining the constraint of equation (5). We are not strict by the type. It will also explore the attributes of different types together. We are denoting second order pseudolikelihood as $pl_2(G_D;\theta)$.

$$pl_2(G_D;\theta) = \sum_{t\in T} \sum_{X^t_{\{i,j\}}\in X^t}^{k} \sum_{v:T(v)=t} p(x^t_{v_{A_i}}|pa_{x^t_{v_{B_i}}}) \sum_{e:T(e)=t} p(x^t_{e_{A_i}}|pa_{x^t_{e_{B_i}}}) \qquad (6)$$

We are considering two dimensions $A_i = \{i,j\}$, which may belong to same or different type of objects. In case when i,j belong to same object , parents will include other dimensions of the same object or same or other dimensions of related objects. When dimension i, j is of different types then parents will include the parent of both objects. Consider the case of citation database, observing paper topic and year will be dependent on other attributes of paper and attributes of author object(related object).

We need to solve the following second order log-pseudolikelihood equation: $\frac{\partial}{\partial\theta}pl_2(G_D;\theta) = 0$. The parameter formed by solving this equation will lead us close to the true estimate. RDN uses non-selective, relational Bayesian classifier or selective, relation probability trees [15]. Relational learners are used to compute the local CPD's necessary to build the model graph. We focus on non-selective model and extend RBC to second order setting.

4 Second Order RBC

Second order RBC employees second order PL function. We assumes *selected set of attribute pairs* to be independent given the class instead of attribute independent assumption. Second order RBC considers selected pair of attributes and deals with the multi-set[5] formed by cross product of individual multi-set of two attributes. Consider in citation network, observing paper month and year, we deal with multi-set of paper month and year respectively. Assume for a given paper, month set contains $\{Jan, Feb\}$ and year contains $\{2002, 2004\}$. Second order RBC makes a multi-set corresponding to this pair of attributes by taking cross product of individual sets (ex. $\{(Jan, 2002), (Jan, 2004), (Feb, 2002), (Feb, 2004)\}$). Since in RBCs, the independent assumption on values of set is experimentally observed to yields best results compared to other methods, we employee the same in second order RBC. Now, we describe the selection of attribute pairs.

Initially we have set of attributes denoted as $A = \{X_1, X_2, \cdots, X_m\}$. Each attribute from set A is associated with a type information, $\forall\ i\ X_i \equiv X^t_i$. These attribute are heterogeneous but second order RBC treats them as homogeneous by ignoring the type information. Second order RBC first makes all possible pair of attributes and denoted as $P = \{\{X_1, X_2\}, \{X_1, X_3\}, \cdots, \{X_{m-1}, X_m\}\}$. Now,

[5] Multi-set is a set whose members can have more than one membership {a, a, a, c}

the second order RBC select elements from P which lead to the full likelihood denoted as set S. Second order RBC does it on the basis of score to reach close to the full likelihood. Equation (7a) show the basic properties of set S. These constraints are in line with that of the composite likelihood dimension sets.

$$S \subseteq P, \text{Such that}$$

$$\forall \; s_i, s_j \; \in \; S, \;\; s_i \cap s_j \; = \emptyset \tag{7a}$$

$$\cup_{i=1}^{|S|} \; s_i = A \tag{7b}$$

5 Construction of set S

Second order RBC constructs S from P. We explore two approaches to make set S, which will lead us closer to full likelihood. Full likelihood function is defined as $P(C|\{a_1, a_2, ..., a_m\})$ where C is the class label.

5.1 Exhaustive Search

Choose the subset, $S \equiv \underset{p \,\subset\, P}{arg\,max} \; P(C|p)$, from P which maximize the likelihood of the class. We have to search for all possible subsets which follows the constraints of Equation (7a) and maximizes the likelihood of the class. Due to computational costs, we look for greedy strategies to arrive at the set S.

5.2 Greedy Approach

Second order RBC uses greedy strategy to make S and assigns a score to each element of P, which is the likelihood of the class given the attribute set (Refer 5.2). $score(p_i) = logP(C|p_i) \;\equiv\; logP(C|\{X_i, X_j\})$. We add maximum score elements of P to S by maintaining the constraints of equation (7a). Second order RBC approximates the full likelihood function using second order PL. According to modified second order PL,

$$P(C|\{X_1, X_2, ...X_m\}) \;\propto\; P(A|C) * P(C) \equiv P(S|C) * P(C) = \prod_{i=1}^{|S|} P(s_i|C) * P(C) \tag{7}$$

Algorithm 1 Assignment of score to set P

```
for $i = 1 \rightarrow |P|$ do
    for all $|Subgraphs|$ do
        $score(p_i) = logP(C|p_i)$
    end for
end for
```

Algorithm 2 Selection of S

```
while $|P|$ do
    $T = max\_score(P)$
    {max_score function selects the maximum score element from set $P$}
    $S = S \cup T$
    remove elements from $P$ where $p_i \cap T \neq \emptyset$
end while
```

Second order RBC predicts the class label assuming subgraphs to be independent. Second order RBC is used to assign the soft labels to the unknown classes of test subgraphs, it is used to give seed to the inference process. We evaluate second order RBC in next section.

Complexity- Second order RBC learning has three major components.

- Assignment of score to all elements of set P, it takes $O(|P|.N)$, where N is number of subgraphs.
- Sorting of the scores, it takes $O(|P| * log(|P|)$.
- Construction of set S takes $O(|P|^2)$

Overall *asymptotic* complexity of second order RBC is $O(|P|.N)$.

6 Experiments and Results

We performed experiments on synthetic and real world data set to demonstrate the situation where second order learning is better idea.

6.1 Synthetic Data Experiments

Experiments in this section demonstrates the comparison of second order PL and PL learning in context of relational dependency networks. First, we use synthetic data to report the improvements due to second order learning in different scenarios. We report effect of attribute correlation in addition to autocorrelation on different data sets.

We generated heterogeneous data subgraphs. Data set contains object of type X, having four binary attributes $x_1, x_2, x_3, class$. We generated the class label given degree distribution of S object, in particular used *normal distribution* with supplied mean and variance [6]. Degree of a vertex represents the number of other vertices it is connected. Once the class label of all X type vertices are generated, we generate all other attributes based on probabilistic relationship with the class, submitted attribute correlations and autocorrelation. We start with autocorrelation 0.5 for this particular set of experiments. V^i represent an

[6] A detailed discussion on data generation can be found at: http://kdl.cs.umass.edu/proximity/documentation/tutorial/ch06s09.html

vertex of data graph, G_D and V^r represents all related[7] vertices.

Data Characteristics

$$\begin{aligned} P({V^i}_{class=+}|{V^r}_{class=+}) = 0.5, \quad P({V^i}_{x_1=1}|{V^i}_{class=+}) = 0.5 \\ P({V^i}_{x_2=1}|{V^i}_{class=+}) = 0.5, \quad P({V^i}_{x_3=1}|{V^i}_{class=+}) = 0.5 \\ P({V^i}_{x_1} = {V^i}_{x_2}|{V^i}_{class=+}) = 0.95, \quad P({V^i}_{x_3=1}|{V^i}_{x_2=1}) = 0.95 \end{aligned} \tag{8}$$

We create different size sample with a little modification in probabilities to compare second order RBC to RBC in terms of likelihood and accuracy. We run Qgraph[8] queries to generate all subgraphs having at least one linked object. We sample the subgraphs in to train and test using random sampling.

1. **Effect of training size** Second order RBC performance increases as training size increase; Gains both in likelihood estimation and accuracy are observed (refer Figure 4, 5). Second order RBC takes larger training data for stabilization.

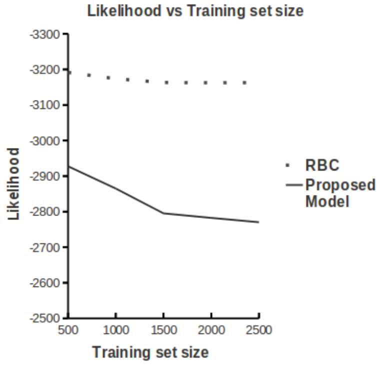


Fig. 4. Effect of training size on Likelihood

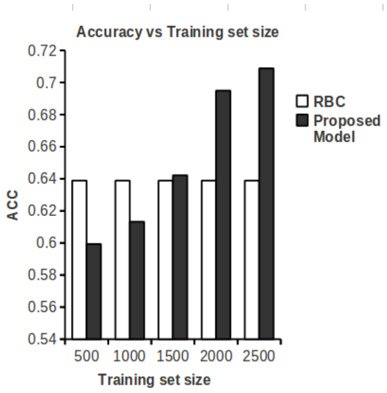


Fig. 5. Effect of training size on ACC

2. **Effect of correlation** We perform experiments to draw the effect of attribute correlation on the performance of model. We can change the correlation among the attributes by modifying the probabilistic relationship given in equation (8). Figure 6 and 7 shows that, our model performs better than existing RBC in highly correlated environment in both likelihood estimation as well as accuracy. Next, we explain the effect of autocorrelation on the model performance.
3. **Effect of autocorrelation** Figure 8 and 9 shows the performance of model in different autocorrelation scenarios. Second order model performs better

[7] Related vertices may contains all nearest neighbours connected in G_D

[8] Qgraph, used by Proximity to make graph queries visual and efficient [11]

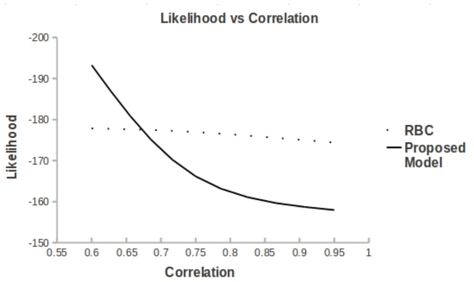


Fig. 6. Effect of correlation on Likelihood

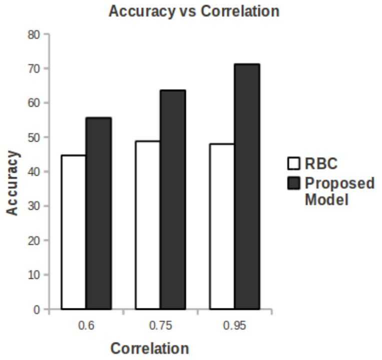


Fig. 7. Effect of correlation on ACC

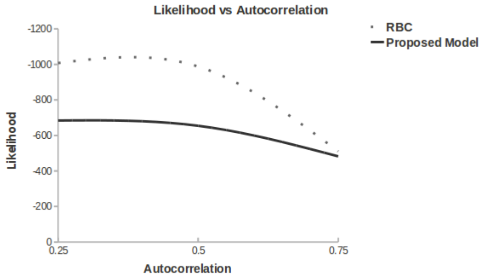


Fig. 8. Effect of correlation on Likelihood

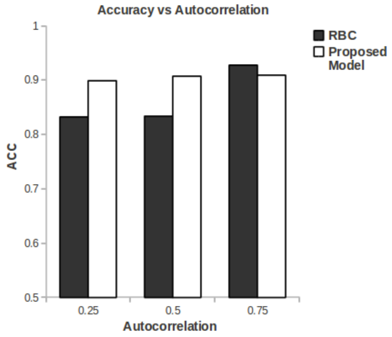


Fig. 9. Effect of correlation on ACC

in low to moderate autocorrelation and comparable in high autocorrelation environment. In high autocorrelation scenarios prediction is biased because same variable of related objects give enough information to predict the attribute.

6.2 Real World Data Experiments

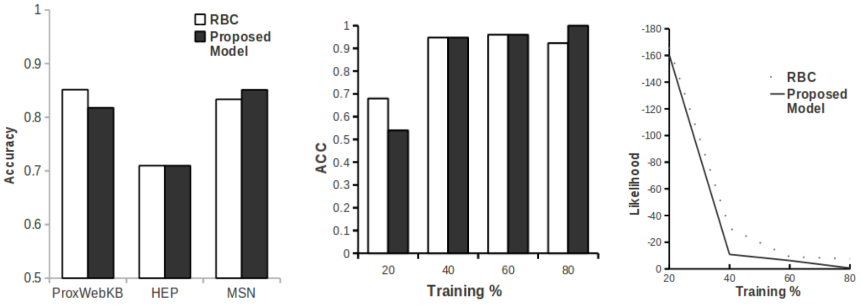
We perform experiments on three real world data sets namely HEP [9], MSN[10] and ProxWebKB[11]. We describe the prediction task in the context of these data sets. All experiments are performed over binary class label. We construct binary class label by discretization of the class label.

[9] The data contains abstract and citation files provided for the 2003 KDD Cup competition, see http://kdl.cs.umass.edu/data/hepth/hepth-info.html

[10] We uses data set available on http://kdl.cs.umass.edu/data/msn/msn-info.html

[11] Data set is taken from CMU Web-KB project from http://www.cs.cmu.edu/ webkb/, modified by Knowledge Discovery Laboratory

ProxWebKB data contains web pages from computer science department of different universities. Web-page can belong to student, course, faculty, research project or staff. We want to predict the category of a page given its linked page categories. MSN is a mobile social network data base. Mote to mote interactions are stored, each mote has its id and links represents trial and time stamp. We want to predict the time stamp of connections. HEP data set is a network of physics papers. We want to predict the topic of paper given paper attributes, author names and publisher.



(a) Accuracy results (b) HEP-Accuracy (c) HEP-Likelihood

Fig. 10. Real world data set results

As shown in Figure 10(a), we observe that our assumption does not fit well on ProxWebKB and HEP data set, but in MSN mote to mote interactions are correlated. To show the strength of our model, we add attributes to the HEP data set which are formed using two or more existing attributes. We want to predict the *acceptability* of a paper which is formed using paper citation degree and journal name. We observe significant improvement in accuracy and likelihood (refer Figure 10(b), 10(c)).

7 Conclusion & Future Work

We have shown the use of second order pseudolikelihood in the RDN learning. We have shown improvements in highly correlated data sets both in parameter estimation and classification accuracy. Our model works well for moderate to large training size. It is true that pseudolikelihood works well in the real world data sets but, we demonstrated the situations where second order learning is a better idea. As we move from PL to FL complexity will increase and some times intractable but by moving to second order we are gaining in performance which is a strong motivation for the use of second order learning in highly correlated environment. In recent times, researchers are trying to optimize the RDN computations and coming up with approaches to make it learn fast [17]. We believe introducing second order pseudolikelihood estimate in combination with fast learn RDN will make it even more faster in case of highly correlated environment.

Acknowledgment

We thank to Knowledge Discovery Laboratory for their useful Proximity tool using which the proposed model is implemented. We would like to acknowledge arXiv archive, Stanford Linear Accelerator Center for HEP data set and Privacy, Internetwork, Security, and Mobile Systems Laboratory at the University of Massachusetts Amherst for MSN data set along with Knowledge Discovery Laboratory, University of Massachusetts Amherst.

References

1. B. C. Arnold and D. Strauss. Pseudolikelihood estimation: some examples. *Sankhya: The Indian Journals of Statistics*, 53:233–243, 1991.
2. J. Besag. Statistical analysis of non-lattice data. In *National Conference on Artificial Intelligence*, volume 24, pages 179–195. JSTOR.org, 1975.
3. Joshua V Dillon and Guy Lebanon. Statistical and computational tradeoffs in stochastic composite likelihood. *Computing Research Repository*, 2010.
4. Joshua V. Dillon and Guy Lebanon. Stochastic composite likelihood. *J. Mach. Learn. Res.*, 9999:2597–2633, December 2010.
5. Peter Flach and Nicolas Lachiche. 1bc: a first-order bayesian classifier. In *Proceedings of the 9th International Workshop on Inductive Logic Programming, volume 1634 of Lecture Notes in Artificial Intelligence*, pages 92–103. Springer-Verlag, 1999.
6. Lise Getoor and Ben Taskar. *Introduction to Statistical Relational Learning (Adaptive Computation and Machine Learning)*. The MIT Press, 2007.
7. B. Gidas. Consistency of maximum likelihood and pseudo-likelihood estimators for gibbs distributions. 10, 1988.
8. David Heckerman, David Maxwell Chickering, Christopher Meek, Robert Rounthwaite, and Carl Kadie. Dependency networks for inference, collaborative filtering, and data visualization. *Journal of Machine Learning Research*, pages 49–75.
9. Manfred Jaeger. Relational bayesian networks. pages 266–273. Morgan Kaufmann, 1997.
10. Manfred Jaeger. Relational bayesian networks. *CoRR*, abs/1302.1550, 2013.
11. Blau Immerman Jensen, H. Blau, N. Immerman, and D. Jensen. A visual language for querying and updating graphs. Technical report, 2002.
12. David Jensen, Jennifer Neville, and Brian Gallagher. Why collective inference improves relational classification. In *In Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 593–598, 2004.
13. Jennifer Neville and David Jensen. Dependency networks for relational data. In *In Proceedings of the 4th IEEE International Conference on Data Mining*, pages 170–177, 2004.
14. Jennifer Neville and David Jensen. Relational dependency networks. *J. Mach. Learn. Res.*, 8:653–692, May 2007.

15. Jennifer Neville, David Jensen, Lisa Friedland, and Michael Hay. Learning relational probability trees. In *In Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 625–630, 2003.
16. Jennifer Neville, David Jensen, and Brian Gallagher. Simple estimators for relational bayesian classifiers. In *Proceedings of the Third IEEE International Conference on Data Mining*, ICDM'03, pages 609–, Washington, DC, USA, 2003. IEEE Computer Society.
17. Oliver Schulte, Zhensong Qian, Arthur E. Kirkpatrick, Xiaoqian Yin, and Yan Sun. Fast learning of relational dependency networks. *CoRR*, abs/1410.7835, 2014.
18. Ben Taskar. Discriminative probabilistic models for relational data. pages 485–492, 2002.
19. Ben Taskar, Pieter Abbeel, Ming-Fai Wong, and Daphne Koller. Relational markov networks, 2007.