Learning from Imbalanced Data in Relational Domains: A Soft Margin Approach

Shuo Yang
School of Informatics and Computing
Indiana University-Bloomington, USA
Email: shuoyang@indiana.edu

Tushar Khot
Department of Computer Science
University Of Wisconsin-Madison, USA
Email: tushar@cs.wisc.edu

Kristian Kersting
Department of Knowledge Discovery
Technical University of Dortmund, Germany
Email: kristian.kersting@cs.tu-dortmund.de

Gautam Kunapuli
UtopiaCompression Corp.
Email: gautam@utopiacompression.com

Kris Hauser
Department of Electrical and Computer Engineering, Duke University, USA
Email: kris.hauser@duke.edu

Sriraam Natarajan
School of Informatics and Computing
Indiana University-Bloomington, USA
Email: natarasr@indiana.edu

***Abstract*—We consider the problem of learning probabilistic models from relational data. One of the key issues with relational data is class imbalance where the number of negative examples far outnumbers the number of positive examples. The common approach for dealing with this problem is the use of sub-sampling of negative examples. We, on the other hand, consider a soft margin approach that explicitly trades off between the false positives and false negatives. We apply this approach to the recently successful formalism of relational functional gradient boosting. Specifically, we modify the objective function of the learning problem to explicitly include the trade-off between false positives and negatives. We show empirically that this approach is more successful in handling the class imbalance problem than the original framework that weighed all the examples equally.**

I. Introduction

Recently, a great deal of progress has been made in combining statistical methods with relational or logical models, in what is now known as *Statistical relational Learning* (SRL) [1]. SRL addresses the challenge of applying statistical learning and inference approaches to problems which involve rich collections of objects linked together in a complex, stochastic and relational world. The advantage of SRL models is that they can succinctly represent probabilistic dependencies among the attributes of different related objects, leading to a compact representation of learned models. While these models are highly attractive due to their compactness and comprehensibility, the problem of learning them is computationally intensive. As with standard graphical models, most of the SRL models have two components: structures (rules) that capture the influences between attributes and parameters (weights or probability distributions) that model uncertainty. Consequently, structure learning problem in SRL is also significantly difficult.

Consequently, structure learning has received increased attention lately, particularly in the case of one type of SRL models called Markov Logic Networks [2], [3]. These approaches provide solutions that are theoretically interesting; however, applicability to real, large applications is nominal due to restricting assumptions in the models and complex search procedures inside the methods. A more recent algorithm called Relational Functional Gradient Boosting (RFGB) [4], [5], based on Friedman's functional gradient boosting [6] addressed this problem by learning structure and parameters simultaneously. This was achieved by learning a set of relational trees for modelling the distribution of each (first-order) variable given all the other variables. The key insight is to view the problem of learning relational probabilistic functions as a sequence of relational regression problems, an approach that was successful in the standard propositional data cases. In turn, it is natural to expect that problems and solutions well known for propositional classification/regression problems can be studied for and transferred to SRL settings. Interestingly, this has not been fully considered so far.

Specifically, imbalanced class distributions have remained as a significant bottleneck to the performance attainable by most standard propositional algorithms. In this paper, we show that this problem is especially prevalent in learning SRL models. This is because, in relational settings, a vast majority of relations between objects are not true, and the number of negative examples far outnumbers the number of positive examples. For instance, consider the relation `Friends(x,y)`. If `x` and `y` can both take 1000 different values, this relation can have 1M different grounded facts with different configurations of x and y. However, in the data base, only less than 50k relations could possibly be true since in real world, most of `Friend` relationships are not true between people. And such imbalance increases as the number of objects in the relations increases. This problem is a critical one in all SRL domains involve relations between people, such as `co-worker`, `advised-by`, `co-authors`, etc. The way Probabilistic Relational Models [7] bypasses this problem is by creating a binary existence variable for every possible relation, but this introduces a similar class imbalance problem while learning the existence variables because this binary relation will be false for all but a very few number of relations.

The approach employed by more successful SRL algorithms such as RFGB is to sub-sample a set of negatives, and use this subset for training. While effective, this method can lead to *a large variance* in the resulting probabilistic model. An alternative method typically is to oversample the minority class but as observed by Chawla [8], this can lead to overfitting on the minority class. This is particularly true for incremental model-building algorithms such as RFGB,

which iteratively attempt to fix the mistakes made in earlier learning iterations. In the propositional domains, this problem is typically addressed by operating and sampling in the feature space [8]. However, in relational domains, the feature vector is not of a fixed length and the feature space can possibly be infinite [9]. Previous work has shown that random sampling of features will not suffice in relational domains [10].

One solution to address the issue of imbalance leading to overfitting in the propositional world is to perform some form of *margin maximization*. A common approach to margin maximization is via regularization, typically achieved via a regularization function [11]–[13]. In propositional and relational functional-gradient boosting methods, common regularization approaches restrict number of iterations, tree size or number of trees learned. While reasonably successful, algorithm behavior is sensitive to parameter selection, and requires additional steps such as cross validation to achieve optimal model tuning. Instead, we explore the use of *cost-based soft-margin maximization*, where a carefully designed cost function relaxes the hard margin imposed by maximum log-likelihood by penalizing certain misclassified examples differently.

Based on the observation that the current RFGB method simply optimizes the log-likelihood by considering the underlying model to be a (structured) log-linear model, we introduce a soft-margin objective function that is inspired by earlier research in log-linear models [14], [15]. The objective is very simple: high-cost examples should be penalized differently from low-cost examples. In our setting, false negatives constitute high-cost examples. This is especially motivated by applications in medical domains (for instance, predicting heart attacks), where it is imperative that any modeling approach reduces the number of false negatives (failed to predict cardiac event with adverse consequences and high cost), even if it comes at the expense of adding *a few more* false positives (predicted a cardiac event that did not happen, moderate cost). The low-cost examples, for this application, would be false positive examples. Our cost function addresses this class imbalance, essentially by placing weights on the false negative and false positive cases. Thus, we propose a soft-margin function, or more specifically, a cost-augmented scoring function that treats positive and negative examples differently.

In this paper we address the *imbalance problem* in relational data by introducing a soft-margin-based objective function. Then, we derive gradients for this soft margin objective function to *learn conditional distributions*. We also show the relation between the soft margin and the current RFGB algorithm. Since the original RFGB algorithm has been extended to learn several types of directed and undirected models [4], [5], [16], our algorithm is broadly applicable and not restricted to a particular class of SRL model. Finally, we evaluate the algorithm on several standard data sets and demonstrate empirically that the proposed framework outperforms standard RFGB in addressing the problem of class imbalance.

II. Background and Related Work

A. Soft Margin Learning for Unbalanced Datasets

Soft margin approaches are a popular approach for unbalanced datasets [17], [18]. Conceptually, our approach is closest to the softmax-margin approach for log-linear models [14] where the objective function is a modified log-likelihood function with a cost function in the normalization term that assigns different weights to different error types (false positives vs false negatives). However, they have not yet been applied to SRL models, which require non-trivial modifications.

Cost-sensitive learning for relational models has been considered by Sen and Getoor [19] where they learn the *parameters* of a conditional Markov network using two cost-sensitive approaches: one based on the expected cost of misclassification and the second based on introducing a cost in the maximum entropy formulation. Our approach learns structure as well as parameters of relational conditional distributions.

B. Relational Functional Gradient Boosting

Friedman [20] proposed a boosting approach where functional gradients are computed for each example over the objective function. These gradients correspond to the difference between the true label and predicted probability of an example and are used to generate a regression dataset. In each iteration, a regression function is learned to fit to these gradients and added into the model to improve the probabilistic predictions. Unlike AdaBoost, this functional gradient boosting (FGB) approach learns a *probabilistic* classifier.

In the relational setting, the functional gradient method has been shown to be successful in learning structure for multiple relational models [4], [5], [16]. Natarajan et al. [4] introduce Relational Functional Gradient Boosting (RFGB) to learn a Relational Dependency Network (RDN) as a set of relational conditional distributions. RFGB represents a probability distribution as a sigmoid over a regression function ψ:

$$P(x|parents(x)) = \frac{\exp \psi(x; parents(x)))}{1 + \exp \psi(x; parents(x)))}. \quad (1)$$

It then calculates the regression values for each grounding of the target predicate by computing the functional gradient of the pseudo-loglikelihood objective function. The gradient $(\frac{\partial \sum_i \log P(x_i)}{\partial \psi(x_i; parents(x_i))})$ for an example x_i is given by

$$\Delta(x_i) = I(x_i = true) - P(x_i|parents(x_i)), \quad (2)$$

where I is an indicator that returns 1 for positive, and 0 for negative examples. The key insight is that the gradients are not summed over all the examples, but instead, are computed for each example. This gradient becomes a weight for that example ($\Delta(x_i)$). In SRL case, this corresponds to computing the gradient (weight) for every grounding. Then, a relational regression tree [21] is learned to fit to these weighted groundings, which is then added to the model. Similar to parametric descent, the sum of these m gradients is the current value of the regression function ψ.

Note that when given an example and the value of all the other groundings in the database, only one path in a tree is satisfied. Then the regression value from the leaf of that path is taken as the gradient and added to the overall gradient. Hence, these different trees can then be summed to obtain the ψ value of the current grounding given its parents. This approach is known as Relational Functional Gradient Boosting (RFGB). This is an iterative procedure where at each step, the gradients are computed for each example and a (relational) tree is learned over all the examples. Then this tree is added to the current

set of trees and the procedure continues till convergence (or when a preset number of trees is learned). But this approach uses a constant cost for type I (false positive) and type II (false negative) errors and can still overfit as more trees are included.

C. Other ensemble methods

AdaBoost [22], the most popular ensemble method, learns a sequence of weak classifiers by reweighting the example after every iteration and has been shown to outperform a single complex model. But boosting often suffers from overfitting after a few iterations [23]. Hastie et al. [12] proposed ϵ-Boost where they regularize by shrinking the contribution of each weak classifier. Jin et al., [24] proposed Weight-Boost, that combines weak classifier with an instance-dependent weight factor. They trade-off between the weak classifier in the current iteration and the classifier based on the previous iterations. Xi et al., [25] minimize a L_1-regularized exponential loss for sparse solutions and early stopping. Rätsch et al., [26] proposed a weight-decay method, where they soften the margin by introducing a slack variable in the exponential loss function. In contrast to our approach, these approaches were applied to propositional non-probabilistic classifiers.

III. SOFT-RFGB

As discussed previously, equation(2) indicates that when a positive (or negative) example is predicted to be true with probability 0 (or 1), the gradient will be 1 (or -1). This gradient ensures that the probability (via the likelihood) of all positive examples is pushed towards 1, and all the negative examples towards 0. While this appears to be reasonable, closer inspection illustrates two key problems with RFGB. First, as we learn more trees, weights are concentrated on outliers, which leads to over-fitting. In each iteration, outliers are misclassified and will have larger gradients; this leads to the algorithm focusing on them in the subsequent iteration resulting in a more complicated model (overfitting). To avoid this, regularization strategies such as limiting the number of trees or tree size are usually implemented. While this is usually effective, this requires careful tuning of parameters.

Second, RFGB treats both positive and negative examples equally. This is evident from the fact that the magnitude of the gradient does not depend on the label. For skewed data sets, we may want to assign higher weights to a particular example class. For instance, as the number of positive examples is very low, we might wish to assign higher weights to them to reduce false negatives. To alleviate this issue, RFGB currently subsamples a smaller subset of negative examples while training. Again, while this is reasonable, the resulting model usually has high variance due to these varying sets of negative examples.

On the other hand, a cost-sensitive approach allows us to address these issues and model the target task more faithfully. This is achieved by adding a cost term that penalizes examples differently. Thus, in addition to simple log-likelihood of the examples, the algorithm also takes into account these additional costs in order to account for the class imbalance. Following the work of Gimpel and Smith [14], we introduce a cost function into the objective, denoted $c(\hat{y}_i, y)$, where $\hat{y}_i$ is the true label of i^{th} instance and y is the predicted label. The new objective is called Soft-RFGB and is defined as:

$$\log J = \sum_i \psi(y_i; \mathbf{X}_i) - \log \sum_{y'_i} \exp\left\{\psi(y'_i; \mathbf{X}_i) + c(\hat{y}_i, y'_i)\right\},$$

where y_i corresponds to a target grounding (a ground instance of the target predicate) of example i with parents $\mathbf{X}_i$. If the cost function is independent of the underlying potential function $\psi(y_i; \mathbf{X}_i)$ to be estimated, the gradient is

$$\frac{\partial \log J}{\partial \psi(y_i = 1; \mathbf{X}_i)} = I(y_i = 1; \mathbf{X}_i) - \frac{P(y = 1; \mathbf{X}_i) e^{c(y_i, y=1)}}{\sum_{y'_i} [P(y'_i; \mathbf{X}_i) e^{c(y_i, y'_i)}]}.$$

We now define the cost function as:

$$c(y_i, y) = \alpha\, I(y_i = 1 \wedge y = 0) \;+\; \beta I(y_i = 0 \wedge y = 1),$$

where $I(y_i = 1 \wedge y = 0)$ is 1 for false negatives and $I(y_i = 0 \wedge y = 1)$ is 1 for false positives. Intuitively, $c(y_i, y) = \alpha$ when a positive example is misclassified, while $c(y_i, y) = \beta$ when a negative example is misclassified. Substituting the cost function in the gradients, we have $\Delta(y_i) =$

$$\begin{cases} 1 - \dfrac{P(y_i = 1; \mathbf{x}_i)}{P(y' = 1; \mathbf{X}_i) + P(y' = 0; \mathbf{X}_i) \cdot e^{\alpha}}, & \text{if } y_i = 1, \\ 0 - \dfrac{P(y_i = 1; \mathbf{x}_i) \cdot e^{\beta}}{P(y' = 1; \mathbf{X}_i) \cdot e^{\beta} + P(y' = 0; \mathbf{X}_i)}, & \text{if } y_i = 0. \end{cases}$$

Defining $\lambda = e^{c(\hat{y}_i, y=1)} \,/\, \sum_{y'} [P(y'; \mathbf{X}_i)\; e^{c(\hat{y}_i, y')}]$, the gradients of the objective function can be rewritten compactly as

$$\Delta = I(\hat{y}_i = 1) \;-\; \lambda P(y_i = 1; \mathbf{X}_i). \tag{3}$$

Our approach works as follow: we iterate through M steps and in each iteration, we generate examples based on the soft-margin gradients. We learn a relational regression tree to fit the examples using FITRELREGRESSIONTREE [4] which is added to the current model. We limit our trees to have maximum L leaves and greedily pick the best node to expand.

To generate the regression examples (Fig. 1 (a)), we calculate the probability of the example being true (p_i) for each example. We then calculate the gradients based on a simplification of (3). The example and its gradient are added to the set of regression examples, S. In the cost-sensitive gradients derived above, the cost parameter λ depends on the parameters α and β. When $\alpha = \beta = 0$, $\lambda = 1 \,/\, \sum_{y'} [P(y'; \mathbf{X}_i)] = 1$, and the original RFGB gradients are recovered; this setting corresponds to ignoring the cost-sensitive term.

For positive examples, we have $\lambda = (P(y' = 1; \mathbf{X}_i) + P(y' = 0; \mathbf{X}_i) \cdot e^{\alpha})^{-1}$. As $\alpha \to \infty$, which amounts to putting a large positive weight on the false negatives, $\lambda \to 0$ and the gradients ignore the predicted probability as $\Delta \to 1$. On the other hand, when $\alpha \to -\infty$, $\lambda \to 1 \,/\, P(y_i = 1; \mathbf{X}_i)$ which means that the gradients are pushed closer to their minimum value of 0 ($\Delta \to 0$). Similarly, for negative examples, we can show that $\beta \to \infty$, then the gradient $\Delta \to -1$, and if $\beta \to -\infty$, then the gradients $\Delta \to 0$. Generally, if $\alpha < 0$ ($\beta < 0$), the algorithm is more tolerant of misclassified positive (negative) examples. Alternately, if $\alpha > 0$ ($\beta > 0$), the algorithm penalizes misclassified positive (negative) examples even more than standard RFGB. Thus, the influence of positive and negative examples on the final learned distribution can be directly controlled by tuning the parameters α and β.

Simply put, if correct classification of positive examples is very important, one can emphasize such importance by assigning positive values to α. If there are many outliers with positive labels which could lead to over-fitting, one can soften the margin by setting $\alpha < 0$, making the algorithm more tolerant. The choice of β has a similar effect on the negative examples. Soft-RFGB *allows flexible adjustments to classification boundaries* in various domains by defining the cost function with two parameters (α and β), which control the gradients of positives and negatives respectively. Returning to our recurring example of clinical classification, we wish to correctly classify as many positives as possible, while at the same time, avoid over-fitting the negatives. In such cases, we set $\alpha > 0$ and $\beta < 0$; we explore these settings in our experiments, which are presented next.

IV. Experiments

A. Domains

We use five standard relational learning domains: Cora, IMDB, Heart Disease, UW and WebKB for empirical evaluation. Table I shows important details of the data sets used.

Domain	Target Predicate	Num of facts	Num of pos	Num of neg	Imb. ratio
Cora	$samebib$	6731	30971	21952	0.71:1
IMDB	$female\_gender$	959	95	173	1.8:1
Heart	num	7453	265	655	2.5:1
UW	$advisedBy$	5039	113	54729	484:1
WebKB	$courseTA$	1912	121	71095	588:1

TABLE I: Details of the experimental domains.

B. Evaluation Metrics

Standard evaluation metrics on relational models include the use of Area Under ROC or PR curves (AUC-ROC or AUC-PR), F_1 score, etc., which measure accuracy with balanced weight between positive and negative examples. Instead, we address domains where misclassification of positive instances (false negatives) costs significantly more than a false alarm (false positives), e.g., in medical diagnosis, security detection, etc. In such domains, the model should identify as many positive cases as possible as long as the precision stays within a reasonable range. To better serve such a goal, we employ evaluation metrics that assign *higher weights to high recall regions*, that is, the top region in an ROC curve [27].

Specifically this paper uses three such metrics: 1) false negative rate, 2) F_5 measure, and 3) weighted AUC-ROC. By comparing the metrics it becomes possible to better understand an algorithm's performance on different parts of the precision-recall curve. For instance, if Algorithm A has lower false negative rate and higher F_5 measure, but similar weighted AUC-ROC to that of Algorithm B, then this suggests that Algorithm A improves on the false negative rate *without sacrificing overall predictive performance*. (Note that false negative rate and F-measure require a classification threshold, this choice will be discussed below.) Details of the latter two metrics will be discussed here.

Our second metric is the F-measure:

$$F_\delta = (1+\delta^2)\frac{Precision \cdot Recall}{\delta^2 \cdot Precision + Recall}, \tag{4}$$

where δ controls the importance of Precision and Recall [28]. Note that F_1 is the harmonic mean of the precision and recall. As $\delta \to \infty$, $F_\delta \to Recall$, and F_δ is recall dominated, while as $\delta \to 0$, $F_\delta \to Precision$, and F_δ is precision dominated. (Although F-measures are typically subscripted with the symbol β, we use δ here to avoid clashing with the parameters used in the cost function of soft-RFGB.) Our paper uses the F_5 metric to increase the importance of recall over precision.

The weighted-AUC measure shifts weight from the bottom regions to the top regions of the ROC curve, in a manner similar to Weng et al. [27]. The vertical dimension of the ROC plot is divided into $N+1$ horizontal strips, and in the x'th section we assign the weight:

$$W(x) = \begin{cases} 1-\gamma, & x = 0, \\ W(x-1) \times \gamma + (1-\gamma), & 0 < x < N, \\ \frac{W(x-1)\times\gamma+(1-\gamma)}{1-\gamma}, & x = N. \end{cases} \tag{5}$$

where $\gamma \in [0,1]$ controls the amount of skewing. Here, $x = 0$ corresponds to the bottom-most area of the ROC curve and is assigned weight $1-\gamma$. Then, weights are transferred recursively for each of the N regions, and γ controls the amount of weight transferred. The metric used in this paper set N=5 and $\gamma = 0.8$. As an aside, our proposed $W(x)$ contains a correction to [27]. To satisfy the conditions in their work, the bottom region must have a weight of $1-\gamma$, and not γ as claimed by [27]. It is important to note that we did **not** change the evaluation metric parameters for different data sets, but rather chose arbitrary parameters so the metrics focus better on false negatives.

C. Results

We consider four popular relational algorithms: (1) The state-of-the-art MLN learning algorithm (denoted as MLN) [5]; (2) A single relational probability tree (denoted as No Boosting) [29]; (3) RFGB as presented in earlier research (denoted as Hard Margin) [4]; and (4) our proposed approach soft-RFGB, with various parameter settings. Table II presents the results.

We performed experiments with $\alpha = 0.5, 1, 2$ and $\beta = -2, -4, -8, -10$. The choice of $\alpha > 0$ reflects a harsher penalty for false negatives, while $\beta < 0$ reflects higher tolerance to false positives. To assign a classification threshold for the false negative rate and F_5 measure, we used the fraction $\#positive/\#(positive + negative)$ for the Cora, Heart, and IMDB datasets. However, for the WebKB and UW data sets, the data is so skewed that this fraction is extremely small which results in classifying every example as positive. To alleviate this, we randomly subsampled the negative examples during testing so that the pos/neg ratio is $1:10$ and used the fraction $1/11$ to calculate the false negative rate and F_5 measure.

We performed four- or five-fold cross validation on each dataset and averaged the evaluation measures discussed above. Using these results, we study the following key questions:
Q1: How does soft-RFGB perform compared to MLN boost?
Q2: How does soft-RFGB perform compared to a single tree?
Q3: How does soft-RFGB perform compared to RFGB ?
Q4: How sensitive is soft-RFGB to the parameter values?

TABLE II: Experimental results.

		MLN	NB	HM	$\alpha = 0.5$				$\alpha = 1$				$\alpha = 2$			
					-2	-4	-8	-10	-2	-4	-8	-10	-2	-4	-8	-10
Cora	WAUC	0.233	0.709	0.723	**0.739**	0.695	0.703	0.703	0.715	0.709	0.707	0.692	0.715	0.696	0.712	0.712
	FNR	0.151	0.136	0.14	0.131	0.072	0.006	0.011	0.131	**0**	**0**	**0**	0.131	**0**	**0**	**0**
	WF	0.832	0.867	0.864	0.872	0.909	0.969	0.965	0.872	**0.974**	**0.974**	**0.974**	0.872	**0.974**	**0.974**	**0.974**
IMDB	WAUC	0.205	0.361	0.361	0.39	0.364	0.383	0.366	0.391	0.393	0.393	0.365	0.368	0.39	**0.395**	0.383
	FNR	0.626	0.663	0.481	0.129	0.118	0.118	0.118	0.118	0.118	**0**	**0**	0.129	**0**	**0**	**0**
	WF	0.367	0.317	0.494	0.823	0.834	0.834	0.834	0.834	0.834	**0.938**	**0.938**	0.823	**0.938**	**0.938**	**0.938**
Heart	WAUC	0.026	0.302	0.295	0.305	0.327	0.34	0.351	0.278	0.295	0.345	0.353	0.281	0.309	**0.353**	0.345
	FNR	0.386	0.641	0.354	0.024	0.005	0.005	**0**	0.02	0.005	**0**	**0**	0.01	**0**	**0**	**0**
	WF	0.569	0.351	0.622	0.894	0.908	0.909	**0.91**	0.896	0.906	**0.91**	**0.91**	0.904	**0.91**	**0.91**	**0.91**
UW	WAUC	0.044	0.765	0.886	0.884	0.907	0.888	0.913	0.89	0.9	**0.914**	0.858	0.91	0.885	0.892	0.892
	FNR	0.047	0.113	0.006	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	WF	0.701	0.877	**0.95**	0.733	0.733	0.733	0.733	0.733	0.733	0.733	0.733	0.733	0.733	0.733	0.733
WebKB	WAUC	0.004	0.484	0.489	0.425	0.47	0.477	0.457	0.453	0.441	0.447	**0.494**	0.44	0.473	0.462	0.466
	FNR	0.43	0.612	0.501	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	WF	0.429	0.378	0.451	**0.742**	**0.742**	**0.742**	**0.742**	**0.742**	**0.742**	**0.742**	**0.742**	**0.742**	**0.742**	**0.742**	**0.742**

First we compare soft-RFGB to boosting MLN (column 3 in Table II) for **Q1**. For all domains, there exists a parameter choice of α and β that significantly decreases the false negative rate; this is especially so for WebKB and UW data sets, where the false negative rate of soft-RFGB reaches zero. Hence, soft-RFGB significantly improves the false negative rate for nearly all values of β. For all domains, soft-RFGB improves on both the weighted AUC-ROC and the F_5 measure for all studied values of α and β. Thus, **Q1** can be answered affirmatively.

To study **Q2** and **Q3**, we turn our attention to the single relational tree (column 4 in Table II) and standard RFGB (column 5). In each domain, soft-RFGB significantly decreases the false negative rate. In all domains except UW, soft-RFGB improves the F_5 measure. This is because our parameter settings for soft-RFGB sacrifice some precision to achieve higher recall; for UW, standard-RFGB *already achieves the highest possible recall* (=1) and soft-RFGB does not have any room to improve recall, though it sacrifices precision, causing F_5 to decrease. Weighted AUC-ROC for soft-RFGB is similar or better than that for RFGB in all domains. Since our goal is to decrease the false negative rate without hurting the overall performance, the results on weighted AUC-ROC strongly suggest that soft-RFGB can achieve better performance than standard RFGB in high recall regions.

Finally, we see that **Q4** is answered affirmatively: *within a reasonably large range of α and β, our algorithm is not sensitive in most domains*. For example, $\alpha > 1$ and $\beta < -2$ produces consistently good performance. This is a major advantage of our cost-sensitive approach over commonly-used tree-regularization approaches, which tend to be highly sensitive to the choice of tree size, or the number of trees. In addition, our parameters α and β have a nice intuitive interpretation: they reflect and incorporate the high costs of misclassifying certain types of examples for real-world domains where this is an important practical consideration.

It is worth noting that although Table II seems to show that the soft-margin performs uniformly better with larger α and β values, this is not always the case. Performance begins to degrade at some point; for example, with $\alpha = 100$, $\beta = -100$, weighted-AUC is only 0.4187 in WebKB domain, compared to 0.4892 for standard RFGB, and 0.5011 for soft-RFGB with the optimal settings of α and β. We observe similar results in all the domains. The point at which performance begins to decline is problem-dependent and worthy of future study.

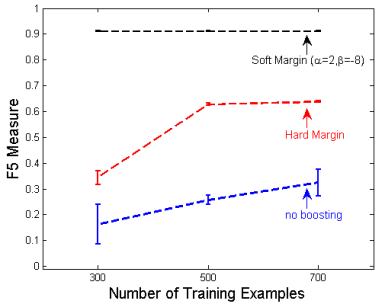
Fig. 1 (b) and (c) show sample learning curves of soft-RFGB, standard RFGB, and a single relational tree, in two domains: Heart Disease and WebKB. We varied the number of training examples and averaged the results over four different runs. Soft-RFGB significantly outperforms the other two algorithms for all the conditions, especially when the number of training examples is small, which signifies that soft-RFGB can efficiently decrease the false negative rate simply via appropriate parameter settings. It can be seen that the two other methods suffer from higher variance than soft-RFGB.

We also performed experiments to evaluate our initial claim that subsampling negative examples to achieve better recall on imbalanced datasets runs the risk of increasing variance of the estimator. Here, we used a re-sampling strategy that sub-sampled the negative examples to match the positive examples (1:1 ratio) during learning. For example, on WebKB, re-sampling using RFGB achieved weighted-AUC of 0.41 compared to 0.48 for RFGB without re-sampling and 0.50 for our new method; re-sampling 0.43, RFGB 0.45 and our new method 0.74 for F_5 measure. We observed similar results in the other domains. The key issue with re-sampling is that sub-sampling negatives may lose important examples, which leads to large variance particularly on highly skewed datasets. To round off the analysis, we also performed over-sampling of positive examples on the heart dataset such that the number of positive and negative examples are equal. The weighted AUC was 0.27 and F_5 was 0.82 when using RFGB with the over-sampled positive examples. To put this in perspective, our algorithm achieved 0.35 AUC and 0.91 F_5 scores. In summary, it can be concluded that soft-RFGB faithfully addresses the class imbalance problem in relational data when compared to any under or over-sampling strategies.

V. Conclusion

We considered the problem of class imbalance when learning relational models and adapted the recently successfully relational functional-gradient boosting algorithms for handling this problem. We introduced a soft margin approach that allowed for the false positives and false negatives to be considered differently. We then derived the gradients (residues) for

```
1: function GenSoftMEGs(Data, F)
2:     S := ∅
3:     for 1 ≤ i ≤ N do
4:         p_i = P(y_i = 1|x_i) = sigmoid(F(y_i; x_i))
5:         if ŷ_i = 1 then
6:             λ = 1/(p_i + (1 − p_i) · e^α)
7:         else
8:             λ = 1/(p_i + (1 − p_i) · e^−β)
9:         end if
10:        Δ(y_i; x_i) := I(y_i = 1) − λP(y_i = 1|x_i)
11:        S := S ∪ [(y_i), Δ(y_i; ; x_i))]
12:    end for
13: return S                    ▷ Return regression examples
14: end function
```



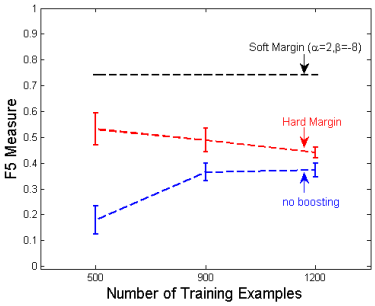


Fig. 1: (a) Function for Generating Examples. (b)Sample learning curve: Heart dataset (c)Sample learning curve: WebKB, Error bars indicate standard errors.

each example using this newly defined optimization function and adapted the original algorithm for learning using these residues. We showed empirically that soft-RFGB minimizes the false negative rate without sacrificing over all efficiency.

This work can be extended in a few interesting directions: first is to understand theoretically the implications of such an approach. While it can be shown that the new optimization function is convex in terms of the original parameter space (the probability distributions), its nature in the functional space is not clear. Understanding this implication is essential to broadly apply this algorithm to several tasks. Second, applying this algorithm to learning more complex models (such as RDNs and MLNs) can lead to interesting insights on the resulting networks. Scaling this algorithm to large relational data such as EHRs where RFGB has been applied earlier is essential. Finally, applying and learning using this approach to label examples in active learning or allow for some unlabeled examples as in semi-supervised learning are interesting and high-impact future research directions.

Acknowledgments

SY, KH and SN gratefully acknowledge National Science foundation grant no. IIS-1343940. SN and TK gratefully acknowledge support of the DARPA DEFT Program under the Air Force Research Laboratory (AFRL) prime contract no. FA8750-13-2-0039. Any opinions, findings, and conclusion or recommendations expressed in this material are those of the authors and do not necessarily reflect the view of the DARPA, AFRL, or the US government.

References

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

[2] S. Kok and P. Domingos, "Learning Markov logic network structure via hypergraph lifting," in *ICML*, 2009.

[3] ——, "Learning Markov logic networks using structural motifs," in *ICML*, 2010.

[4] S. Natarajan, T. Khot, K. Kersting, B. Guttmann, and J. Shavlik, "Gradient-based boosting for statistical relational learning: The relational dependency network case," *Machine Learning*, 2012.

[5] T. Khot, S. Natarajan, K. Kersting, and J. Shavlik, "Learning markov logic networks via functional gradient boosting," in *ICDM*, 2011.

[6] J. H. Friedman, "Greedy function approximation: A gradient boosting machine," *Annals of Statistics*, pp. 1189–1232, 2001.

[7] L. Getoor, N. Friedman, D. Koller, and A. Pfeffer, "Learning probabilistic relational models," *Relational Data Mining, S. Dzeroski and N. Lavrac, Eds.*, pp. 307–338, 2001.

[8] N. Chawla, "Data mining for imbalanced datasets: An overview," in *Data Mining and Knowledge Discovery Handbook*, 2010, pp. 875–886.

[9] D. Jensen and J. Neville, "Linkage and autocorrelation cause feature selection bias in relational learning," in *ICML*, 2002.

[10] T. Khot, S. Natarajan, and J. Shavlik, "Relational one-class classication: A non-parametric approach," in *AAAI*, 2014.

[11] B. Saha, G. Kunapuli, N. Ray, J. Maldjian, and S. Natarajan, "Ar-boost: Reducing overfitting by a robust data-driven regularization strategy," in *ECML/PKDD (3)*, 2013, pp. 1–16.

[12] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*. Springer, 2001.

[13] R. Schapire and Y. Freund, *Boosting: Foundations and Algorithms*. The MIT Press, 2012.

[14] K. Gimpel and N. Smith, "Softmax-margin crfs: Training log-linear models with cost functions," in *HLT-NAACL*, 2010.

[15] F. Sha and L. Saul, "Large margin hidden markov models for automatic speech recognition," in *NIPS*, 2006.

[16] S. Natarajan, S. Joshi, P. Tadepalli, K. Kristian, and J. Shavlik, "Imitation learning in relational domains: A functional-gradient boosting approach," in *IJCAI*, 2011.

[17] Z. Liu, "Maximal-margin approach for cost-sensitive learning based on scaled convex hull," in *ISNN*, 2011.

[18] M. Kim, "Large margin cost-sensitive learning of conditional random fields," *Pattern Recognition*, vol. 43, no. 10, 2010.

[19] P. Sen and L. Getoor, "Cost-sensitive learning with conditional markov networks," in *ICML*, 2006.

[20] J. Friedman, "Greedy function approximation: A gradient boosting machine," *Annals of Statistics*, vol. 29, 2001.

[21] H. Blockeel and L. D. Raedt, "Top-down induction of first-order logical decision trees," *Artificial Intelligence*, vol. 101, pp. 285–297, 1998.

[22] Y. Freund and R. Schapire, "Experiments with a new boosting algorithm," in *ICML*, 1996.

[23] D. Mease, A. Wyner, A. Buja, and R. Schapire, "Boosted classification trees and class probability/quantile estimation," *Journal of Machine Learning Research*, vol. 8, p. 2007, 2006.

[24] R. Jin, Y. Liu, L. Si, J. Carbonell, and A. Hauptmann, "A new boosting algorithm using input-dependent regularizer," in *ICML*, 2003.

[25] Y. Xi, Z.Xiang, P. Ramadge, and R. Schapire, "Speed and sparsity of regularized boosting," in *AISTATS*, 2009, pp. 615–622.

[26] G. Raïsch, T. Onoda, and K. Müller, "An improvement of adaboost to avoid overfitting," in *NIPS*, 1998.

[27] C. G. Weng and J. Poon, "A new evaluation measure for imbalanced datasets," in *Seventh Australasian Data Mining Conference (AusDM 2008)*, ser. CRPIT, J. F. Roddick, J. Li, P. Christen, and P. J. Kennedy, Eds., vol. 87. Glenelg, South Australia: ACS, 2008, pp. 27–32.

[28] C. J. V. Rijsbergen, *Information Retrieval*, 2nd ed. Butterworth-Heinemann, 1979.

[29] J. Neville, D. Jensen, L. Friedland, and M. Hay, "Learning Relational Probability trees," in *KDD*, 2003.