Why Stacked Models Perform Effective Collective Classification

Andrew Fast and David Jensen
University of Massachusetts Amherst
Department of Computer Science
140 Governors Drive, Amherst MA 01002
{afast,jensen}@cs.umass.edu

Abstract

Collective classification techniques jointly infer all class labels of a relational data set, using the inferences about one class label to influence inferences about related class labels. Typical collective classification schemes use computationally-intensive iterative algorithms or approximate joint inference techniques. Kou and Cohen recently introduced an efficient relational model based on stacking that, despite its simplicity, performs equivalently to more sophisticated joint inference approaches. This stacked relational model trains on the inferred labels of related instances, instead of the true labels which are not typically present at inference time. This permits the use of efficient exact inference in place of more computationally-intensive approximate joint inference. There are at least two possible causes for the unexpected high performance of the stacked approach: a reduction in inference bias (resulting from training on inferred rather than true labels) or a reduction in inference variance (due to the use of exact rather than approximate inference). Using experiments on both real and synthetic data, we show that the primary cause for the performance of the stacked model is the reduction in bias from learning the stacked model on inferred labels rather than the true labels. The reduction in variance due to conditional inference also contributes to the effect but it is not as strong. In addition, we show that the performance of the joint inference and stacked learners can be attributed to an implicit weighting of local and relational features at learning time.

1 Introduction

Collective classification approaches for relational data use the relations among entities to jointly infer class labels of the entire dataset. Because relational data often capture autocorrelation relationships, i.e., correlation between the same attribute on different entities, inferences about one entity are able to influence inferences about other entities. For example, the topics of research articles and their citations are often correlated. Making an inference about the topic of one paper provides substantial evidence about the topics of cited papers.

Collective classification is often framed as a joint inference problem over the data. Consequently, current approaches for collective classification utilize iterative algorithms or approximate joint inference techniques such as Gibbs sampling or loopy belief propagation [24]. These joint inference approaches can be difficult to implement, require many iterations to converge, and can be difficult to determine whether they have converged [24].

Recently, Kou and Cohen [11] demonstrated that collective classification using a stacked relational model was sufficient to achieve comparable results to existing approaches using joint inference . This is both surprising and provocative as stacking is easy to implement, efficient, and avoids the challenge of determining the convergence of approximate inference techniques. The stacked approach described by Kou and Cohen uses a non-relational base model to produce inferred class labels on related instances. The stacked relational model is trained on these inferred labels rather than the true labels. This approach has two advantages. First, by using the inferred labels the stacked model can avoid the "bait and switch" experienced by other

relational learners when the true labels of related instances are available during learning but not available at inference time. Second, since the inferred labels are always available during inference, the stacked relational model can use efficient exact inference techniques, avoiding additional variability and computational expense of joint inference approaches.

Using experiments on both real and synthetic data, we show that the primary cause for the performance of the stacked model is the reduction in bias from learning the stacked model on inferred labels rather than the true labels. The reduction in variance due to conditional inference also contributes to the effect but it is not as strong. In addition, we show that the performance of the joint inference and stacked learners can can be attributed to an implicit weighting of local and relational features at learning time. The stacked model weights local information more highly than existing approaches, which mitigates common sources of error during inference such as the absence of class labels on related instances. We apply a recently developed bias/variance framework for collective classification [20] to better understand bias and variance trade-offs made by each algorithm .

2 Background and Related Work

2.1 Collective Classification

Collective classification techniques are designed to make inferences in relational domains where the labels to be predicted are interrelated. These interrelations allow inferences about one instance to influence inferences about related instances. Collective approaches have been shown to yield higher accuracies than non-collective models particularly when the labels of related instances exhibit autocorrelation [8]. Common applications of collective classification include classification of linked documents such as webpages, patents, or scientific documents [1, 17, 25]; and bioinformatics domains such as gene function prediction [19].

Most solutions to these applications view collective classification as a joint inference problem, simultaneously inferring the class label on every instance. Sen et al. [24] provide a summary of successful collective classification algorithms including iterative algorithms [12, 15, 16], relaxation labeling approaches [1, 13], Gibbs sampling [17, 19], and belief propagation [25].

In this work, we compare the performance of the stacked model with the relational dependency network (RDN) for collective classification [19]. An RDN is a pseudolikelihood model consisting of a collection of conditional probability distributions (CPDs) that have been learned independently from data. Typically, the CPDs used in a dependency network are represented by probability estimation trees, although any conditional model suffices. Although the CPDs are learned independently, they represent a coherent joint probability distribution over the variables in the data when learned from a sufficiently large training set [6]. The RDN performs inference using Gibbs sampling, an approximate inference technique that relies on repeated sampling from conditional distributions.

2.2 Stacking

Kou and Cohen [10, 11] recently introduced a new approach, based on stacking, that does not rely on joint inference for collective classification. Stacking is a type of ensemble method that uses the outputs of one classifier as inputs to another classifier [26]. A stacked relational model allows inferences about one instance to influence inferences about other instances but uses a different mechanism than other approaches to collective classification. Rather than using joint inference to propagate inferences among instances, the stacked model described by Kou and Cohen uses a non-relational model to predict the class label for each instance and uses those inferred labels on related instances as input to a stacked relational model[11].

The learning algorithm used by Kou and Cohen to learn the stacked relational model is shown in Figure 1. First, Kou and Cohen learn a non-relational classifier (S_0) as a base model to predict the labels of the instances using only intrinsic features. This base model is then used to infer labels for each of the instances. To avoid any bias from applying the base model on the same data from which it was learned, stacked learners use a cross-validation-like procedure to train the base model [11, 26].

Given a training set D, a learner B, a set of instances $\mathbf{x}$ and a relational template T:

1. Learn the base model S_0 on instances $\mathbf{x}$.
2. For $k = 1...K$:
   (a) Construct cross-validated labels using model S_{k-1}.
   (b) Construct $\mathbf{x_k} = \mathbf{x_k} \cup T(\mathbf{x_{k-1}})$. That is, augment $\mathbf{x}$ with predicted labels of related instances.
   (c) Learn the stacked model S_k using $\mathbf{x_k}$.

Figure 1: The stacked learning algorithm of Kou and Cohen.

Next, new features are created using a *relational template* T which can be any function that aggregates the class labels of related instances. Kou and Cohen use a relational template that returns the count of each value of the related class labels. This augmented feature set is then used to learn a stacked relational model using both intrinsic and relational features as input. In general, a stacked model can include many layers learned in this fashion, with each subsequent layer using the predicted class labels produced by lower levels. Kou and Cohen show that a single layer of stacking with a relational template is sufficient for collective classification [11].

At inference time, the base model is applied to the test set to create the inferred class labels, then the stacked relational model is applied to produce the final predictions. Since the stacked model is learned on the inferred labels, all input features are known at inference time permitting exact inference. In contrast, other collective models learn on the true labels, which are not known at inference time. This requires more costly iterative approximate inference techniques.

3 Experimental Design

Kou and Cohen [11] demonstrated that a stacked learner produced models with similar accuracy to a relational dependency network (RDN) [19] on collective classification tasks in relational domains. This is a surprising result as the stacked learner differs from the RDN in both learning and inference. At learning time, the RDN trains using the true labels of related instances whereas the stacked learner does not. Instead, the stacked learner uses (possibly noisy) inferred labels on related instances generated by a base learner that considers only intrinsic features. At inference time, however, the true labels of related instances in the test set are not typically fully known. To address this, the RDN uses state-of-the art approximate joint inference to collectively infer labels for the test instances. The stacked learner does not require the true labels of related instances, again relying on the inferred labels generated by the base model. Despite ignoring potentially useful information during learning and not using sophisticated joint inference techniques, the stacked learner is still able to perform as well as the RDN.

To understand these surprising results, we explore the errors made by the stacked learner and the RDN. The error of predictive models can be decomposed into bias and variance components, and there is a trade-off between these two components when learning a model from data [3, 5, 7]. Considering a larger model space via additional parameters can decrease bias but often leads to increased variance. Many simple models have been shown to perform quite well despite higher bias due to a decreased variance component [4]. For collective classification, Jensen et al. [9] showed that relational models require a parameter space that is only incrementally larger than a similar non-relational model. The errors of relational models have only a slightly larger variance component but often a substantially smaller bias component when relational information is available.

In collective classification tasks, the inference procedure can be an additional source of both bias and variance of an algorithm [20]. Inference bias and variance arises from both the use of non-deterministic approximate inference techniques [20] and the strength and availability of relational information at inference time which may vary across test sets [13, 15, 20]. Previous formulations of bias and variance, which assume an exact inference method, only measure the bias and variance of an algorithm due to the learning. The overall error of collective classification algorithms is the sum of the bias and variance due to both learning and inference.

RDNs and stacked learners construct different types of models, and thus exhibit different bias and variance trade-offs between learning and inference. In the case of collective classification, RDNs model relationships among true class labels. In contrast, stacked learners produce both a base model of the relationships between the intrinsic attributes and the class label and a relational stacked model of relationships among the output of the base models (inferred labels) and the actual class labels. To lack learning bias, a stacked learner would have to either produce unbiased models at both levels, or produce stacked models that compensate for the bias of the base models. A more likely scenario is that stacked models would have higher learning bias due to initial bias in learning the base model and that bias being amplified by the stacked model. Indeed, experiments reported in Section 4 indicate that stacked models do exhibit higher learning bias.

To obtain similar overall performance to the RDN, the stacked learner must offset its increased learning bias with a decrease in inference bias, inference variance, or both. The stacked learner and the RDN have two differences which could account for the decrease in the inference loss of the stacked learner. First, by not relying on the true labels of related instances, the stacked learner avoids a potential source of inference bias when the true labels are absent. Second, by using exact inference in place of approximate joint inference, the stacked learner avoids a potential source of inference variance. In the following sections, we describe experiments on both real and synthetic data designed to determine the contributions of bias and variance to the error of each algorithm and explore how the stacked learner is able to achieve equivalent performance to the RDN.

3.1 Learning Algorithms

To determine whether the performance of the stacked learner can be attributed to reductions in inference bias or inference variance, we compared five different learning algorithms that vary in both the learning and inference components. The algorithms are summarized in Table 1. The first model, called Base, learns using only intrinsic features and, therefore, cannot consider the class labels of related instances. When considered alone, the Base model serves as a lower bound on the performance of collective classification. The second model, called Stacked, is based on the stacked learner described by Kou and Cohen [11]. Stacked uses the feature set of the Base model augmented with an additional feature indicating the count of the values of the inferred class labels on related instances. The inferred class labels are generated by the predictions of the Base learner alone. More details on the Stacked model can be found in Section 2.2. Both the Base and Stacked models are able to use exact inference techniques.

We also evaluated the performance of a relational dependency network (RDN) model[19]. The RDN considers a similar feature set as the Stacked model. However, in place of the inferred labels, the RDN uses a feature indicating the count of the values of the true labels of related instances. For inference, the RDN uses Gibbs sampling, an approximate joint inference approach.

To determine how the choice of inference procedure affects the Stacked algorithm, we created a new algorithm, Stacked-Gibbs, which differs from Stacked only in the type of inference used. Stacked-Gibbs uses approximate joint inference with Gibbs sampling in place of exact inference. By comparing the performance of the Stacked-Gibbs model with both the Stacked and RDN models, we can determine whether the performance of Stacked can be attributed to a decrease in inference bias or to a decrease in inference variance.

To provide an upper bound on the performance of collective classification, we considered an oracle algorithm which provides a probabilistic ceiling on both the Stacked and RDN models. This model, called Ceiling, is given the true labels on related instances during inference. This additional knowledge allows the

Table 1: Summary of the models used in the experiments. Related labels indicates the availability of the labels on related instances during learning. Inference types are either exact inference with a conditional model or approximate relational joint inference using Gibbs sampling. The ceiling model indicates the probabilistic ceiling of models if the true labels were available during both learning and inference.

	Learning	**Inference**	
Learner	Related Labels	Type	True Labels Available
Base (Intrinsic Only)	-	Exact	-
Stacked	Inferred	Exact	No
Stacked-Gibbs	Inferred	Approx.	No
RDN	True	Approx.	No
Ceiling	True	Exact	Yes

use of exact inference techniques, which eliminates all sources of both inference bias and inference variance.

3.2 Implementation

For the algorithms described above, it is sufficient to learn a single conditional distribution of the class label given the available features. This model can then either be applied using exact (conditional) inference or approximate (relational) joint inference. We implemented these five models using code for relational probability trees (RPTs) and relational dependency networks (RDNs) found in PROXIMITY 4.3[1], an open-source system for knowledge discovery in relational domains. The RPT is a probability estimation tree for relational data [21]. It uses a χ^2 statistic and a significance threshold of 0.05 to determine whether to add a split to the tree during learning. We used the default implementation of both the RPT and RDN with the following modifications. Although other aggregations of related instances are provided, we limited the RPT to only consider COUNT aggregations to replicate the experimental settings of Kou and Cohen [11]. For efficiency reasons, we limited the maximum depth of the RPTs to five. All experiments requiring joint inference used 100 iterations of Gibbs sampling with no burn in period. Our convergence results (not reported here) confirmed the conclusion of Kou and Cohen who showed that 100 iterations of Gibbs sampling is sufficient for convergence of the Gibbs chain [11]. We chose the RPT in place of of the MaxEnt learner used by Kou and Cohen because the RPT is a selective model and we wanted to confirm that the desirable performance reported by Kou and Cohen using the MaxEnt classifier was due to stacking and not an artifact of the MaxEnt classifier itself.

4 Experiments with Synthetic Data

To explore the bias/variance trade-offs of the STACKED and RDN models, we generated a series of synthetic datasets that varied along three dimensions: the strength of relational information (autocorrelation), the strength of local information (intrinsic correlation), and the proportion of labeled data appearing in the test set. Using these synthetic datasets, we then computed the bias and variance for both the learning and inference components of the algorithms. These experiments confirmed our intuition that the STACKED model has larger learning bias than the RDN model. These experiments also showed that this increase in learning bias is offset by a reduction in both inference bias and inference variance; however, the overall comparison depends on the characteristics of the data.

[1]See http://kdl.cs.umass.edu/proximity.

4.1 Synthetic Data Generation

We considered simulated relational data that consists of a two-dimensional lattice of nodes with each node connected to its four immediate neighbors. To ensure that all nodes in the graph are equivalent we added an additional "frame" of nodes around the periphery of lattice. These additional nodes are not considered in the loss computation but their attribute values are used during learning and inference as related values to the core nodes.

To generate the attributes of the nodes in the lattice, we used the collective data generation process described in the earlier work of Jensen et al. [9]. Autocorrelated binary class labels on the nodes were generated in a two-step process. We first assigned each node a random class label C from the distribution $P(C = 1) = 0.5$. We then used 200 iterations of Gibbs sampling to produce the final class labels. The parameters of the model used for Gibbs sampling were manually generated to produce the desired level of autocorrelation in the data. The levels of autocorrelation we considered are summarized in Table 2.

Once the class labels were generated, we generated three intrinsic attributes A_1, A_2, A_3. The value of the attribute A_1 was drawn from a distribution that depends on the class label. This distribution was based on $P(C|A_1)$ and specifies the strength of intrinsic attributes. The values we considered are shown in Table 2. The values of the remaining intrinsic attributes A_2 and A_3 were assigned randomly according to $P(A_i) = 0.5$.

Table 2: Parameters of synthetic data generation. Datasets were generated for all possible combinations of parameters.

Parameter	**Possible Values**
Training Set Size	1024 objects
Test Set Size	484 objects
Autocorrelation	{0.01, 0.27, 0.49, 0.75}
$P(C\|A_1)$	{0.6, 0.75, 0.9}
Proportion Labeled	{0.0, 0.3, 0.6,0.9}

4.2 Computing Bias and Variance

Squared loss between a true label t and a prediction y for a particular instance x is defined as $L(t, y) = (t-y)^2$. Traditional bias/variance analysis computes the expected loss of an example over models learned on different training sets. If an approximate inference technique is used, inference can also contribute to the overall bias and variance of the model, so loss must be computed from the expectation over both learning and inference. Intuitively, the inferences of a single learned model produced by approximate inference techniques vary both across multiple datasets and across multiple runs of inference on the same dataset.

Following the definitions in Neville and Jensen [20] , learning bias is defined as

$$B_L(x) = (E_L[t] - E_L[y])^2$$

which represents the loss of the optimal prediction $E_L[t]$ and the mean prediction over learning sets $E_L[y]$. Learning variance can be defined as

$$V_L(x) = E_L[(E_l[y] - y)^2$$

This represents the average loss of each instance when compared to the mean prediction over learning sets. Inference bias and variance can be defined similarly:

$$B_I(x) = (E_L[y] - E_{LI}[y])^2$$

$$V_I(x) = (E_{LI}[(E_L[y] - y)^2]) - (E_L[(E_{LI}[y] - y)^2])$$

1. For each outer trial $i = 1...5$:
   (a) Generate test i; record optimal predictions.
   (b) For each learning trial $j = 1...5$:
      i. Generate training set j.
      ii. Learn BASE, STACKED, and RDN models of C on training set j.
      iii. Infer marginal probabilities for test set i with true labels, record *learning* predictions.
      iv. For each inference trial $k = 1 \ldots 5$ and proportion labeled $p = \{0.0, 0.3, 0.6, 0.9\}$:
         A. Label p% of test set with the true label.
         B. Infer marginal probabilities for unlabeled test instances; record *total* predictions.
         C. Measure squared loss.
   (c) Calculate *learning* bias and variance from distributions of *learning* predictions. Calculate *inference* bias and variance from distributions of *total* predictions.
2. Calculate average model loss, average *learning* bias/variance, average *inference* bias/variance.

Figure 2: Pseudocode for computing bias/variance estimates (modified from Neville and Jensen [20]).

where $E_{LI}[y]$ represents the mean prediction averaged over both learning and inference.

The total loss is a composition of both learning and inference error components where γ is a bias interaction term:

$$E_{LI}[L(t, y)] = N(x) + B_L(x) + B_I(x) + \gamma + V_L(x) + V_I(x)$$

We followed the procedure outlined in Neville and Jensen [20] and described in Figure 2 to compute the bias and variance of the each of the models.

4.3 Analysis

The learning bias results of the RDN and STACKED models shown in Figure 3 confirm that the learning bias is the largest component of error for the STACKED model. The learning bias of the STACKED MODEL varies with both autocorrelation and intrinsic correlation, though the autocorrelation is the stronger effect. This highlights the inability of the STACKED model to fully utilize related class labels due to the added noise of the inferred labels on related instances. In contrast, the RDN can take full advantage of the additional relational information much more efficiently, leading to decreased learning bias as autocorrelation increases beyond moderate levels.

The inference bias of the STACKED model is much less than the inference bias of the RDN as shown in Figure 4. In data with high autocorrelation, an error on a single instance can contribute to many errors on related instances, leading to highly biased predictions when the learned model weights related information highly. This effect is exaggerated in our synthetic data as there are very few additional features to help guide inference. Typically, real data contain many different variables which can help smooth the predictions and mitigate the bias of a few costly errors.

The inference variance of the RDN is larger than the inference variance of the of the STACKED model (see Figure 5). This was to be expected since the RDN uses Gibbs sampling to perform inference. Since

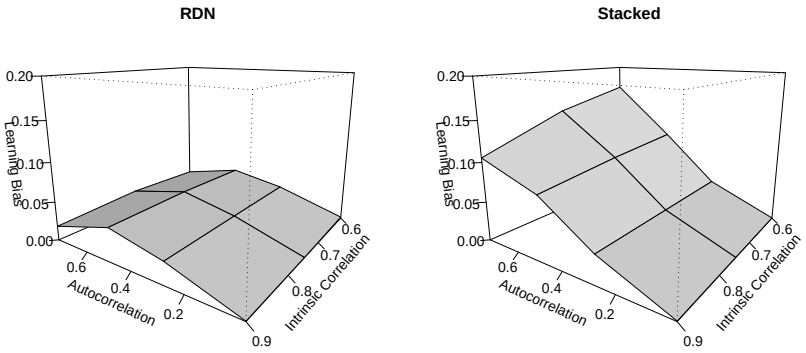

Figure 3: Learning Bias of both the Stacked and RDN models as autocorrelation and intrinsic correlation vary.

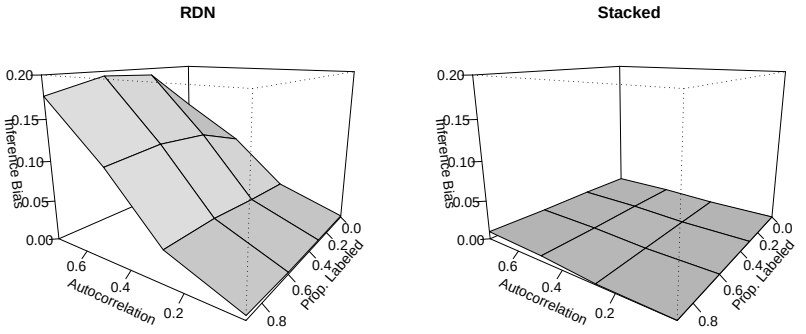

Figure 4: Inference bias of both the Stacked and RDN models as autocorrelation and proportion of the test set labeled vary. Intrinsic correlation is fixed at 0.6.

Gibbs sampling is an approximate approach, the final predictions can vary over multiple runs of inference. Though it also uses Gibbs sampling for inference, the Stacked-Gibbs model has inference variance that is only slightly larger than the inference variance of the Stacked model. This can be attributed to an interaction effect between learning and inference of the Stacked and Stacked-Gibbs models. The inferred labels used by the Stacked and Stacked-Gibbs models are predicted using intrinsic features, which do not vary across iterations of the Gibbs sampler. In contrast, the RDN relies on the relational information which is sampled from the Gibbs sampler, leading to higher variability in the final results.

These synthetic data experiments were designed to demonstrate the bias/variance trade-offs embodied by each of the algorithms. As such, they provide confirmation that the Stacked model differs from the RDN model in both inference bias and learning bias but these experiments may not accurately reflect the range of performance of real data. In addition to autocorrelation and intrinsic correlation, real data may vary with additional characteristics such as the pattern of linkage among instances, latent group structures, and the size of the training and test sets [20].

5 Experiments with Real Data

We compared the performance of the learned algorithms described in Section 3.1 on four binary collective classification tasks drawn from real-world relational data sets. To provide a cost-insensitive evaluation, we

evaluated the tasks using area under the ROC curve (AUC) averaged across the cross-validation folds. These results indicate that the desirable performance of the STACKED model can be primarily attributed to learning on inferred labels. Learning on inferred labels implicitly weights the the intrinsic features more highly than the relational features at learning time. This allows the STACKED models to avoid an increase in inference bias when the true labels are not available at inference.

5.1 Learning Tasks

The first classification task was to predict the topic of machine learning papers appearing the Cora dataset. We used a sample of 4330 computer science papers from Cora, a database of research articles extracted automatically from the web [14]. Our sample contained information about authors, publication venue, and author institution as well citation relations between papers. Unlike Kou and Cohen, we did not consider features of the full text of the articles. We used 10-fold cross validation on the Cora data.

The second classification task was to predict whether the localization of a Yeast gene was in the nucleus. The data were drawn from the Gene dataset, made available for the 2001 KDD Cup [2]. The data contain 1243 genes and 1734 interactions between genes. We used 10-fold cross-validation for the Gene data.

The third classification task was to predict whether a movie grossed more than $2 million in its opening weekend. Movies and related entities were drawn from the Internet Movie Database (IMDb) [3]. We considered a sample of 1382 movies released in the United States between 1996 and 2001. Along with the movies, we considered all actors, directors, producers, and studios associated with the movie. To allow for collective classification, we included links between movies through actors, directors, and studios. We used 5-fold cross validation on the IMDb.

The fourth classification task was to predict whether a webpage belongs to a student. The pages were spidered from four computer science departments and were released as part of the WebKB data [2]. We used 4-fold cross-validation by department.

5.2 Results

The STACKED algorithm produced models with area under the ROC curve (AUC) that were significantly higher ($p < 0.01$) than the RDN models in all datasets (shown in Figure 6). All significance tests were performed using a one-sided, paired t-test on the set of AUC scores from each training fold. In the Cora and Gene datasets, the differences between the models can be attributed to the higher inference variance of the RDN and do not represent substantial differences. On the IMDb data, the stacked model significantly outperformed the RDN model ($p < 0.01$). This can be explained by a difference in the weighting of intrinsic and relational features at learning time. We describe this weighting in more detail below. These experiments provide a comparison of two different learners. To facilitate this comparison the features provided to each learner were restricted to a subset of the possible features, therefore, the results reported here may not be indicative of the maximum possible performance achievable by either algorithm.

Exact inference also reduces error as the STACKED-GIBBS approach performs significantly worse ($p < 0.05$) than the STACKED model in every dataset except WebKB. This gap indicates the increased inference variance inherent in approximate inference approaches.

Each of the datasets demonstrate different weightings of intrinsic and relational features, as shown in Table 3. These weights are computed by following the path from the root node of the learned relational probability tree model to the leaf used to classify each instance. The weight of each instance is the fraction of its path that considers features on related class labels. For example, if a path to a leaf passes through one feature on a related class label and three intrinsic features, then the weight of the instances classified in that leaf is 0.25. The RPT is a selective model, only a subset of the possible features appear in each tree. In every dataset, the RDN weights relational features more highly than intrinsic features. This is expected

[2] See http://www.cs.wisc.edu/ dpage/kddcup2001/.
[3] See http://www.imdb.com.

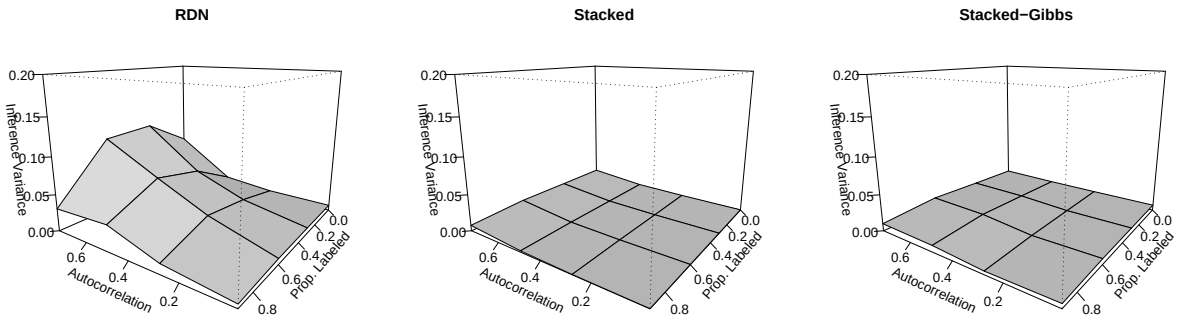

Figure 5: Inference variance of RDN, **Stacked** and **Stacked-Gibbs** models as autocorrelation and proportion of the test set labeled vary.

behavior as the RDN trains on the true value of related class labels, while the STACKED model trains on noisier inferred labels.

Table 3: Weights of relational features in the **Stacked** and RDN models. Weights represent the fraction of each prediction that can be attributed to a feature on a related class label and are computed using the path in the tree used to classify each instance.

Data Name	RDN	STACKED
Cora	0.93	0.80
Gene	0.33	0.21
IMDb	0.58	0.29
WebKB	0.12	0.04

In the Cora data, relational features are weighted highly by both the RDN and STACKED models while intrinsic features provide little or no value to the models. This is also demonstrated by the gap in performance between the base model and ceiling model, which was applied on a fully labeled test set (Figure 6). Although the stacked model does not rely on the true class label of related instances, the local information is not useful and the inferred class label produced by the base model is uninformative. Due to the noisy base model, STACKED would be unable to fully capitalize on the related class labels if they were present.

On the IMDb data, the RDN weights relational features almost twice as heavily as it does as intrinsic features, the largest difference in weights of any dataset. Since the true labels of related instances are not present in the test set and the RDN relies strongly on those labels, the RDN does not perform well. The STACKED learner is better able to capitalize on the local information during inference to significantly outperform the RDN.

The results on the Gene and WebKB datasets highlight two other performance regions. In Gene, the relational information does not provide much additional information as is evidenced by the small gap between the base and ceiling models. Although the relational features have some weight, they do not contribute much to the final score. In contrast, in WebKB the relational features have low weights but provide enough information to allow both the RDN model and the STACKED model to outperform the base model although the difference are not significant at $p < 0.05$.

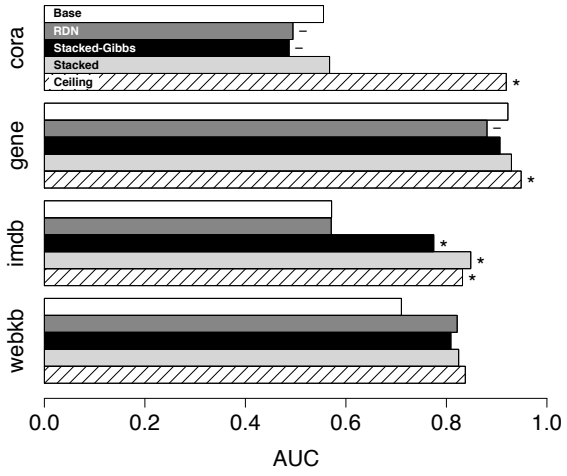

Figure 6: Area under the ROC curve (AUC) for collective classification tasks. Asterisks beside a bar indicate a significant increase in AUC over the base model ($\mathbf{p} < 0.01$) using a one-sided, paired t-test. Dashes beside a bar indicate a significant decrease.

6 Discussion

Relational stacked models using a single level of stacking are able to perform collective classification tasks as well as state-of-the-art methods employing approximate inference techniques, such as the RDN, on both real-world and synthetic data. Our results confirm the results of Kou and Cohen that a single iteration of stacking produces equivalent performance to 100 iterations of Gibbs sampling [11]. We show that a stacked learner employing a relational template exhibits higher learning bias when compared to an RDN, but the stacked learner also exhibits lower inference bias and lower inference variance. This improvement is due to the reliance of the stacked learner on inferred labels of related instances.

By relying on inferred labels instead of the true labels, the stacked learner weights intrinsic features more highly than features on related class labels. This helps the stacked learner mitigate the "bait and switch" effect that occurs when relational models train on the true labels only to find those labels absent at inference, a primary source of error in the RDN. In addition, the presence of inferred labels allows the stacked learner to use exact inference in place of approximate inference with Gibbs sampling, an additional component of error of the RDN.

This use of stacking is similar to multiple imputation models for the analysis of large survey data with missing values [22, 23]. Imputation approaches typically consider two models: an imputation model and an inference model. The imputation model is used to fill in missing data with plausible values, which are then used as inputs for the inference model. Stacked models use the base model to impute the class labels on related instances. The second-level stacked model then is able to use the inferred labels of related instances.

This process of imputation used by stacking makes a decision about the relative weighting of local information and the class labels of related instances during learning. Stacking determines this weighting implicitly, based on the quality of the base model. The inferred labels present a noisy signal of the true labels used by the RDN. The added noise reduces the desirability of features on the related class labels at learning time, leading to increased learning bias. This suggests an opportunity to explore alternative weightings of relational and intrinsic features to see if it is possible to reduce learning bias within a stacking framework. In addition, it may be beneficial to consider ensembles of stacked models each with different weighting.

The surprising performance of the stacked model does not come without cost, however. Due to the

cross-validation-like procedure used by the STACKED model during learning, the longer runtime of learning often negates the dramatic efficiency improvements due to the use of exact inference. The potential for the efficiency improvements of the STACKED model will depend on the type of problem being considered. If the RDN requires many iterations of Gibbs sampling before convergence, then the additional cost of learning the STACKED model may be warranted. Also, if inference on many different data sets is expected, then the additional cost of learning the STACKED model may be offset by the savings in inference time over many datasets.

The RDN and STACKED models each represent a single point in the space of bias/variance tradeoffs. Other approaches for collective classification, such as relational Markov networks [25] and latent group models [18], exhibit different trade-offs under different data conditions [20]. By varying other data characteristics such as the linkage among instances, we can further explore the effectiveness of stacked models for collective classification.

7 Acknowledgments

This material is based on research sponsored by the Air Force Research Laboratory and the Intelligence Advanced Research Projects Activity (IARPA), under agreement number FA8750-07-2-0158. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. The views and conclusion contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the Air Force Research Laboratory, the Intelligence Advanced Research Projects Activity, or the U.S. Government.

Portions of this analysis were conducted using Proximity, an open-source software environment developed by the Knowledge Discovery Laboratory at the University of Massachusetts Amherst.
(http://kdl.cs.umass.edu/proximity/)

References

[1] S. Chakrabarti, B. Dom, and P. Indyk. Enhanced hypertext classification using hyperlinks. *Proc. ACM SIGMOD*, 1998.

[2] M. Craven, D. DiPasquo, D. Freitag, A. McCallum, T. Mitchell, K. Nigam, and S. Slattery. Learning to extract symbolic knowledge from the World Wide Web. In *Proceedings of the 1998 National Conference on Artificial Intelligence*, 1998.

[3] P. Domingos. A unified bias-variance decomposition for zero-one and squared loss. In *Proceedings of the Seventeenth National Conference on Artificial Intelligence*, pages 564–569, 2000.

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

[5] S. Geman, E. Bienenstock, and R. Doursat. Neural networks and the bias/variance dilemma. *Neural Computation*, 4(1):1–58, 1992.

[6] D. Heckerman, D. Chickering, C. Meek, R. Rounthwaite, and C. Kadie. Dependency networks for inference, collaborative filtering, and data visualization. *The Journal of Machine Learning Research*, 1:49–75, 2001.

[7] T. Heskes. Bias/Variance Decompositions for Likelihood-Based Estimators. *Neural Computation*, 10(6):1425–1433, 1998.

[8] 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. Morgan Kauffman, 2002.

[9] D. Jensen, J. Neville, and B. Gallagher. Why collective inference improves relational classification. In *Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2004.

[10] Z. Kou. Stacked graphical learning. Technical Report CMU-ML-07-123, Machine Learning Department, School of Computer Science, Carnegie Mellon University, 2007.

[11] Z. Kou and W. W. Cohen. Stacked graphical models for efficient inference in Markov random fields. In *Proceedings of the 2007 SIAM Conference on Data Mining (SDM)*, 2007.

[12] Q. Lu and L. Getoor. Link-based text classification. In *Proceedings of the International Joint Conference on Artificial Intelligence*, 2003.

[13] S. Macskassy and F. Provost. Classification in networked data: A toolkit and a univariate case study. *Journal of Machine Learning Research*, 8:935–983, May 2007.

[14] A. McCallum, K. Nigam, J. Rennie, and K. Seymore. Automating the construction of Internet portals with machine learning. *Information Retrieval*, 3(2):127–163, 2000.

[15] L. K. McDowell, K. M. Gupta, and D. W. Aha. Cautious inference in collective classification. In *Proceedings of the 22nd AAAI Conference on Artificial Intelligence*, 2007.

[16] J. Neville and D. Jensen. Iterative classification in relational data. In *Proc. AAAI-2000 Workshop on Learning Statistical Models from Relational Data*, pages 13–20, 2000.

[17] J. Neville and D. Jensen. Collective classification with relational dependency networks. In *Workshop on Multi-Relational Data Mining (MRDM-2003)*, 2003.

[18] J. Neville and D. Jensen. Leveraging relational autocorrelation with latent group models. In *Proceedings of the 5th IEEE International Conference on Data Mining*, 2005.

[19] J. Neville and D. Jensen. Relational dependency networks. *Journal of Machine Learning Research*, 8, 2007.

[20] J. Neville and D. Jensen. A bias-variance decomposition for collective inference models. *Machine Learning Journal*, forthcoming.

[21] 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.*, 2003.

[22] D. Rubin. *Multiple Imputation for Nonresponse in Surveys*. Wiley-Interscience, 2004.

[23] J. Schafer. Multiple imputation: a primer. *Statistical Methods in Medical Research*, 8(1):3, 1999.

[24] P. Sen, G. M. Namata, M. Bilgic, L. Getoor, B. Gallagher, and T. Eliassi-Rad. Collective classification in network data. Technical Report CS-TR-4905, University of Maryland, College Park, 2008.

[25] B. Taskar, P. Abbeel, and D. Koller. Discriminative probabilistic models for relational data. In *Proceedings of Uncertainty in Artificial Intelligence*, 2002.

[26] D. Wolpert. Stacked generalization. *Neural Networks*, 5(2):241–259, 1992.