Classification from One Class of Examples for Relational Domains

Tushar Khot and **Sriraam Natarajan*** and **Jude Shavlik**
University of Wisconsin-Madison, {tushar,shavlik}@cs.wisc.edu
* Indiana University, natarasr@indiana.edu

Abstract

One-class classification approaches have been proposed in the literature to learn classifiers from examples of only one class. But these approaches are not directly applicable to relational domains due to their reliance on a feature vector or a distance measure. We propose a non-parametric relational one-class classification approach based on first-order trees. We learn a tree-based distance measure that iteratively introduces new relational features to differentiate relational examples. We update the distance measure so as to maximize the one-class classification performance of our model. We also relate our model definition to existing work on probabilistic combination functions and density estimation. We experimentally show that our approach can discover relevant features and outperform three baseline approaches.

1 Introduction

Traditional classification approaches rely on labeled examples from all classes to learn a model. But in several real world problems, labeled examples are available for only one class. For example, in information extraction from text, a common annotation approach is to mark the true instances in the text (UzZaman et al. 2012; Kim et al. 2009). Since human annotation in these tasks can be quite costly, the negative examples are not annotated or in rare cases very few negative examples are annotated. For instance, presidents of countries will be marked in an article for training, there is no explicit mention of what the negative examples of presidents are. One possible approach to handle this problem is to assume that the unmarked instances are negative examples for the task at hand. Since the number of unmarked examples can be very large, this can increase the learning time for standard classification approaches. A bigger problem is that this can possibly lead to class imbalance where the number of instances of the negative class can dominate the positive instances. Finally, this simplistic assumption could be incorrect as all instances of positive examples may not be annotated (e.g., all mentions of presidents need not be annotated).

Consequently, a lot of research has focused on developing one-class classification (OCC) methods (Khan and Madden 2009) that directly learn using examples from only one class or using very few examples of the other class. For example, Tax and Duin (1999) fit a hypersphere with the smallest radius around the labeled examples. All the examples inside this hypersphere are assumed to belong to the labeled class. Anomaly/Novelty detection tasks can also be viewed as an OCC task where many examples are available for the normal class and few or no examples of the anomalous class are provided. One-class classification can also be viewed as a specific case of semi-supervised learning (Zhu and Goldberg 2009) with the labeled data being provided for only one class. Hence methods such as clustering and nearest-neighbors have also been used previously for OCC (de Ridder, Tax, and Duin 1998). These methods, in general, rely on a feature space representation (e.g., to fit a hypersphere) or distance measure (e.g., for clustering) to perform OCC.

While successful, these methods rely on a propositional representation of the data (i.e., as a flat feature vector). Data in the real world is inherently relational i.e., they consist of inter-related objects and dependencies. Such structured data cannot be easily represented using a fixed-length feature vector and doing so can result in loss of possible information (Jensen and Neville 2002). Recently, several algorithms have been developed that exploit the rich structure while handling uncertainty. Collectively called Statistical Relational Learning (Getoor and Taskar 2007) (SRL), these algorithms combine rich representations such as first-order or relational logic with probabilistic models.

While representations such as first-order logic provide a lossless way to model structured data, they introduce the additional complexity of having unbounded features associated with each example. E.g., consider the task of predicting the blood type of a person based on his lineage. We can potentially introduce a feature for the bloodtype of every ancestor of a person. As a result, standard distance metrics such as Euclidean norms cannot be naively used for relational problems. One potential approach for relational OCC is performing propositionalization followed by using the standard OCC techniques. Standard propositionalization may pick irrelevant features and as a result OCC techniques on these propositionalized datasets can underperform, as we show empirically. The propostionalization approach needs to select the features so as to optimize the one-class classification performance which can be computationally expensive with standard OCC methods.

We propose a non-parametric approach for OCC using a tree-based distance measure between relational examples which is learned by directly optimizing the OCC performance. We define this measure using the path similarity in relational decision trees. We combine the distances from multiple incrementally learned trees and labeled examples to calculate the probability of an example belonging to the labeled class. We present three interpretations of our approach: 1) based on combining rules for probability distributions (Natarajan et al. 2008), 2) based on using our distance with Kernel Density Estimation (Bishop 2006) and 3) based on using a similarity kernel with Support Vector Data Description (SVDD). Using this model definition, we derive a scoring method that allows for learning the trees so as to maximize the likelihood of the labeled examples. Although we use unlabeled examples in our approach to discover the relevant features, we do not require them to be specified (which we explain in Section 3). Another interesting interpretation of our approach is to view the method as a *feature selection* strategy for standard OCC methods. From this perspective, our approach can be seen as a way to propositionalize relational data for OCC methods.

We make several key contributions in this work: 1) We present a new distance metric for relational examples that can be learned incrementally. 2) We propose a relational one-class classification approach using this distance metric. 3) We relate and place this work in the context of existing work on combination functions, existing work on density estimation and OCC. 4) We show how we can perform feature selection for propositional OCC methods by evaluating our trees. 5) Finally, we demonstrate over multiple domains that our approach can learn more accurate models than standard relational learning and OCC methods.

2 Related Work

One-class classification (OCC) problem was first described by Moya et al. (1996) for an automatic target recognition task using neural networks. Following their work, Scholkopf et al. (2000) learn a Support Vector Machine (SVM) to separate the labeled examples from the origin. Tax and Duin (1999), on the other hand, learn a hypersphere with the smallest radius that encompasses the labeled examples. They have also shown that using the Gaussian kernel with a normalized dataset gives solutions comparable to the SVM approach. DeComite et al. (1999), on the other hand, use modified C4.5 to perform one-class classification. All these approaches rely on a propositional representation.

Anomaly/outlier detection methods can also be viewed as a one-class classification approach. Although there have been many outlier detection methods proposed in literature (Chandola, Banerjee, and Kumar 2009), outlier detection approaches generally assume that outliers are far and isolated (Liu, Ting, and Zhou 2012). This is not true for the one-class classification problem where the labeled examples can be clustered and close to other unlabeled examples. Semi-supervised learning approaches can also be used for OCC as shown by Ridder et al. (1998). Our approach can be viewed as semi-supervised relational learning as it leverages unlabeled examples for learning the relational distance function.

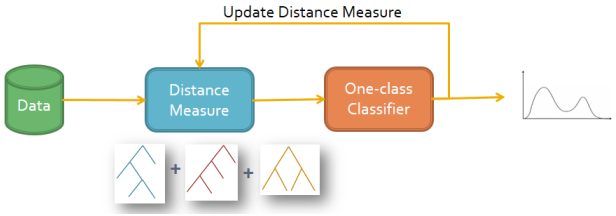

Figure 1: Relational one-class classification system design

One of the earliest works on relational one-class classification (Muggleton 1997) learns the smallest first-order hypothesis that can cover the labeled examples. Since this work only performed Boolean predictions, it was extended to perform relational density estimation (Cussens 1998). This approach relied on calculating proportions of examples captured by every hypotheses which can be prohibitively expensive for large domains.

Multiple propositionalization techniques and distance measures have been proposed for relational data which can potentially be used for OCC. One simple unsupervised approach to generate propositional features is by creating Boolean features corresponding to random first-order logic clauses (Anderson and Pfahringer 2008). Relational Instance-based Learning (RIBL) (Horváth, Wrobel, and Bohnebeck 2001) uses similarity between local neighborhoods to calculate the similarity between examples. Such unsupervised approaches may not capture long range information while we leverage the labeled examples to guide our approach. kFoil (Landwehr et al. 2010) uses a *dynamic* approach to learn clauses to propositionalize relational examples for SVMs. Each clause corresponds to one Boolean feature and is scored based on the improvement in the SVM learned from the propositionalized data. Rather than learning a new model for every candidate structure, we compute the error reduction on the fly locally.

3 Relational OCC

The high-level overview of our approach to Relational One-Class Classification (RelOCC) is shown in Figure 1. Our approach consists of the following steps:

1. Learn a first-order tree for calculating relational distances. (Section 3.1)
2. Use the distance function to perform OCC. (Section 3.2)
3. Learn the next distance tree to improve the classification. (Section 3.3)
4. Go to Step 2.

Table 1 presents the terminology used throughout the paper. We will now describe the individual steps of our approach.

3.1 Tree-based distance

Inspired by semantic similarity measures (D'Amato, Staab, and Fanizzi 2008), we propose a distance measure based on the paths taken by examples in a relational decision tree. For

Term	Description
x, x_i, y	Relational examples
$\{x_1, x_2, \ldots x_l\}$	Labeled examples
$lca(x_1, x_2)$	Lowest Common Ancestor of the leaves reached by x_1 and x_2
$depth(n)$	Depth of node n. $depth(Root) = 0$.
$d_i(x, y)$	Distance between x and y based on i^{th} tree
$D(x, y)$	Distance between x and y based on all trees
$P(y \in \mathcal{L})$	Probability of y being in labeled class
α_j	Weight of labeled example x_j
β_i	Weight of i^{th} tree
$I(condition)$	Indicator function that returns 1 if condition is true, 0 otherwise.
$P^t(y \notin \mathcal{L})$	$P(y \notin \mathcal{L})$ using t trees

Table 1: Description of terms used in this paper.

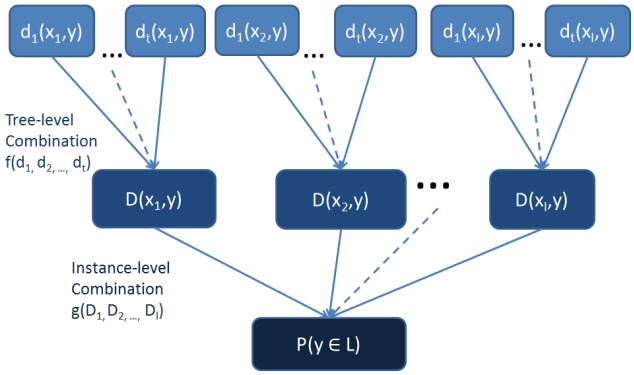

Figure 2: Density Estimation using Distance measure

instance, consider two examples that reach the same node, n in a decision tree before taking separate paths down the tree i.e., n is the lowest common ancestor (LCA) of these examples. If the node n is at depth of three, the two examples share *at least* three common attribute values. If n is at depth of one, they share at least one attribute value. Intuitively, the two examples in the first case are more likely to be similar than in second case. Thus, our distance measure is inversely proportional to the depth of LCA of two example paths.

$$d(x_1, x_2) = \begin{cases} 0 & lca(x_1, x_2) \text{ is a leaf} \\ e^{-\lambda \cdot depth(lca(x_1, x_2))} & \text{otherwise} \end{cases}$$

Examples that differ at the root node i.e. have no common attribute will have a distance of 1. A major feature of this distance definition is that it can be used on any off-the-shelf tree learner (such as TILDE (Blockeel and Raedt 1998)). The key difference from a first-order decision tree is that, in our formulation, there are no values associated with the leaves since all we are interested in are the paths that determine the distance between examples. The parameter λ (set to 0.5 in experiments) ensures that the distance value decreases gradually as the depth increases.

Properties It can be easily shown that this distance function satisfies the following distance metric properties:

- $d(x_1, x_2) \geq 0$ (non-negative)
- $d(x_1, x_2) = d(x_2, x_1)$ (commutative)
- x_1 and x_2 are identical $\Rightarrow d(x_1, x_2) = 0$[1] (equality)
- $d(x_1, x_3) \leq d(x_1, x_2) + d(x_2, x_3)$. (triangle property)
  Proof: If $l_{ij} = depth(lca(x_i, x_j))$, $l_{13} \geq \min(l_{12}, l_{23})$. WLOG, $l_{12} = \min(l_{12}, l_{23})$.
  $l_{13} \geq l_{12} \Rightarrow e^{-l_{13}} \leq e^{-l_{12}} \Rightarrow e^{-l_{13}} \leq e^{-l_{12}} + e^{-l_{23}} \Rightarrow d(x_1, x_3) \leq d(x_1, x_2) + d(x_2, x_3)$. This can also be proven if x_1 and x_3 reach the same leaf node i.e., $d(x_1, x_3) = 0$. ■

Using just one tree may not suffice when dealing with potentially infinite dimensions in relational data. A simple next step would be to learn multiple relational trees. However this step introduces an additional complexity –the need for a way to combine these trees without increasing the complexity of the already complex problem. Our solution is inspired by density estimation techniques that we present next.

[1]The reverse condition does not hold.

3.2 Density Estimation Model

We now define a model (shown in Figure 2) to combine distances from multiple trees and then distances from multiple examples to perform relational one-class classification. Recall that to calculate the probability estimate of an example y to belong to the labeled class, we use the distances between y and the labeled examples. We denote the distance between the current example y and a labeled example x from the i^{th} tree as $d_i(x, y)$. $D(x, y)$ combines the tree distances to return the overall distance between x and y. We finally compute the density estimate $P(y \in \mathcal{L})$ using the distances from all the labeled examples $\{x_1, x_2, \ldots x_l\}$. The combination function that we use in both levels is weighted mean:

$$D(x_j, y) = \sum_i \beta_i d_i(x_j, y) \text{ s.t.} \sum_i \beta_i = 1, \beta_i \geq 0 \quad (1)$$

$$P(y \notin \mathcal{L}) = \sum_j \alpha_j D(x_j, y) \text{ s.t.} \sum_j \alpha_j = 1, \alpha_j \geq 0 \quad (2)$$

One of the key advantage of this function is that it allows for efficient learning of trees as we show in Section 3.3. Also, this approach is closely related to existing research in one-class classification thus making it theoretically well grounded. We first present the learning of the trees followed by the relation to existing methods.

3.3 Model Learning

Tree Learning As shown in Figure 1, we update the distance measure by iteratively adding to the set of trees. Assume that we have learned t trees already and are now learning the $(t+1)^{th}$ one.We can use any off-the-shelf tree learner and we employ the method described by TILDE tree learning (Blockeel and Raedt 1998). Since our distance metric only relies on the LCA position, we can make local scoring decisions at every node. If a pair of examples are already split by the tree, any further splits at lower levels will have no impact on the distance between these examples i.e., adding

a node to the tree only affects the distance between the pair of examples that are split at that node. Specifically, splitting example x_i and x_j at node n, increases their distance $d_{t+1}(x_i, x_j)$ from zero to $\Delta_{t+1}(x_i, x_j) = e^{-depth(n)}$.

We use a modified splitting criterion which is the squared error over the examples i.e., $\sum_y [I(y \notin \mathcal{L}) - P(y \notin \mathcal{L})]^2$. $I(y \notin \mathcal{L})$ is the indicator function which returns 1 if y is an unlabeled example. Hereafter we use $I(y)$ to compactly represent $I(y \notin \mathcal{L})$. As can be seen here, our scoring function does rely on using the unlabeled examples in the dataset. Even if unlabeled examples are not provided, these can be generated by using constants of matching types (e.g. $\{friends(x, y) | \forall x, y \in People, friends(x, y) \notin \mathcal{L}\}$). Relying on only the labeled examples for learning a distance function will result in trivial distances of $d(x, y) = 0$.

Consider $\mathbf{x}$ to be the set of examples that reach node n. $\mathbf{x_l}$ and $\mathbf{x_r}$ are the set of examples that take the left and right branch respectively. Since introducing the node has no impact on the distances (and in turn probabilities) for examples not reaching node n, the squared error is

$$\begin{aligned}
&\sum_{y \in \mathbf{x}} [I(y \notin \mathcal{L}) - P(y \notin \mathcal{L})]^2 \\
&= \sum_y [I(y) - \Sigma_j \alpha_j \{\Sigma_i^t \beta_i d_i(x_j, y) + \beta_{t+1} \Delta_{t+1}(x_j, y)\}]^2 \\
&= \sum_y \left[I(y) - P^t(y \notin \mathcal{L}) - \Sigma_j \alpha_j \beta_{t+1} \Delta_{t+1}(x_j, y)\right]^2
\end{aligned}$$

At every node in the tree, we minimize this squared error. If y is close to the labeled examples ($P^t(y \notin \mathcal{L}) \approx 0$) for an unlabeled example ($I(y) = 1$), increase in the distances to the labeled examples will reduce the squared error. If y is already spread apart from the labeled examples (i.e. $P^t(y \notin \mathcal{L}) \approx 1$), the distances to the labeled examples will not be increased any further. As a result, minimizing the squared error *introduces new features* to spread out the examples and *prevents redundant features* being added.

Since introducing node n only increases the distance between the pair of examples split at the node, $\Delta_{t+1}(x_j, y) \neq 0$ for $x_j \in \mathbf{x_l}, y \in \mathbf{x_r}$ or $x_j \in \mathbf{x_r}, y \in \mathbf{x_l}$. Hence the splitting criterion can be modified to minimizing

$$\begin{aligned}
&\sum_{y \in \mathbf{x_r}} \left[I(y) - P^t(y \notin \mathcal{L}) - \Sigma_{x_j \in \mathbf{x_l}} \alpha_j \beta_{t+1} \Delta_{t+1}(x_j, y)\right]^2 \\
&+ \sum_{y \in \mathbf{x_l}} \left[I(y) - P^t(y \notin \mathcal{L}) - \Sigma_{x_j \in \mathbf{x_r}} \alpha_j \beta_{t+1} \Delta_{t+1}(x_j, y)\right]^2
\end{aligned}$$

We use a greedy search procedure for learning the tree where we pick the feature whose split reduces this squared error. The key insight is that the use of the mean squared error along with our relational path based distance function allows us to efficiently make these local scoring decisions. Note that our approach is non-parametric as the only parameter is the number of trees that increases as more data is obtained. Thus we can potentially update the distances in online fashion.

Weight Learning We set uniform weights on the examples and trees initially. After learning each tree, we update the weights α and β for weighted mean function. We use a co-ordinate gradient descent approach where we iteratively update α and β to minimize the squared error. Since there is no closed form solution, we update the weights only after learning an entire tree instead of updating after learning every node in the tree. The gradient of error w.r.t. α is

$$\begin{aligned}
&\frac{\partial}{\partial \alpha_j} \sum_y \left[I(y) - \Sigma_j \alpha_j \Sigma_i \beta_i d_i(x_j, y)\right]^2 \\
&= -2 \sum_y \left[I(y) - P(y \notin \mathcal{L})\right] \Sigma_i \beta_i d_i(x_j, y)
\end{aligned}$$

and the gradient w.r.t. β is

$$-2 \sum_y \left[I(y) - P(y \notin \mathcal{L})\right] \Sigma_j \alpha_j d_i(x_j, y)$$

To ensure that weights are always greater than 0 and sum to 1, we project them as described by Natarajan et al. (2008).

3.4 Model Interpretations

We now place our work in context to existing work on learning parameters in SRL and propositional OCC methods.

Combining Rules: Since $d_i(x, y) \leq 1$, it can be interpreted as the probability of example x being not equal to y based on the i^{th} tree, i.e., $P_i(x \neq y) = d_i(x, y)$. Similarly $D(x, y)$ can be viewed as overall probability of example x being not equal to y, i.e., $P(x \neq y) = D(x, y)$. This can be viewed as using two-level combining rules (Natarajan et al. 2008) in directed SRL models with the weighted mean combination function. At first level, we combine probabilities from the set of trees (Eqn. 1). At second level, we combine probabilities based on individual examples (Eqn. 2).

Kernel Density Estimation: An alternate interpretation of our model is that of using Kernel Density Estimation (Bishop 2006) with a triangular kernel[2] and bandwidth 1.

$$\begin{aligned}
1 - \sum_j^n \alpha_j \sum_i \beta_i d_i(x_j, y) &= \sum_j^n \frac{1}{n} (1 - \sum_i \alpha_j \beta_i d_i(x_j, y)) \\
&= \sum_j^n \frac{1}{n} |1 - D'(x_j, y)| = \sum_j^n \frac{1}{nh} K(\frac{D'(x_j, y)}{h}) \text{ for h=1}
\end{aligned}$$

where $D'(x_j, y) = \sum_i \alpha_j \beta_i d_i(x_j, y)$. Since the term $|x_j - y|$ cannot be defined easily for relational examples, we use D' function to represent the absolute distance needed by the kernel density estimator. The value of D' is always below 1 and as a result is within the bounds of the triangular kernel.

SVDD: SVDD (Tax and Duin 1999) approach learns a hypersphere around the labeled one-class examples. An example z is classified as belonging to the labeled class if $(z \cdot z) - 2 \sum_i \alpha_i (z \cdot x_i) + \sum_{i,j} \alpha_i \alpha_j (x_i \cdot x_j) \leq R^2$. Since the classification only depends on the dot product, SVDD also uses kernel functions. Using the function $1 - D$ in our case as the kernel functions[3] changes the function used for classification to $1 - D(z, z) - 2 \sum_i \alpha_i (1 - D(z, x_i)) + \sum_{i,j} \alpha_i \alpha_j (1 - D(x_i, x_j)) \leq R^2$. Since $D(z, z) = 0$, we can rewrite the decision boundary as $1 - \sum_i \alpha_i D(z, x_i) \geq$

[2] $K(x, y) = (1 - |x - y|) \cdot I(|x - y| < 1)$

[3] this is possible as kernel functions are similarity functions

$1 - \sum_i \alpha_i D(x_k, x_i)$[4] where x_k is any labeled example. We use the left hand side of the equation as the probability estimate for z belonging to the labeled class. Our definition allows to return probabilistic estimates, but we can now also use $1 - \sum_i \alpha_i D(x_k, x_i)$ as a decision boundary to predict the example class. Since we optimize a different function, it is not necessary for all examples x_k to result in the same boundary. But we can approximate it by using the average distance to all labeled examples.

Feature Selection for OCC: It is also possible to view our proposed approach as a feature selection strategy for standard one-class classification methods. More specifically, we can convert every path from root to leaf into a Boolean feature since these paths can be represented as a horn clause. Since the resulting feature is Boolean, doing this for every path will yield a propositionalized data set that can then employ standard OCC methods such as SVM-OCC. We call this approach SVMOCC-Rel and show the importance of this feature selection strategy in our experiments. Propositionalization of relational data can possibly result in infinite features as a walk through the relational graph can lead to several resulting features. For instance, when reasoning about a particular genetic disposition, it is possible to use generations of ancestors' information. Our method on the other hand, can be seen as selecting these relational features (i.e., identifying the most important set of ancestors) for a propositional classifier making it possible for the propositional methods to work with relational data more seamlessly.

Implementation details: Our approach begins with the distance of 0 between every pair of examples. During every tree learning step, we use only a subsample of examples for efficiency which has shown to improve performance in highly skewed datasets (Chan and Stolfo 1998). We further subsample the examples based on their current predictions. Based on active learning (Settles 2012) techniques, we pick unlabeled examples based on their proximity to the decision boundary as well as the misclassified examples. For the weight learning step, we use a step length of $\eta = 0.001$.

4 Experiments

We evaluate the following different algorithms:

- RelOCC: This approach is our relational OCC method using the predictions from the combination function.
- RND: We modify the tree learner to randomly pick features at every node. To ensure a strong baseline, we do not pick any feature that results in more than 95% examples going to one branch.
- O-Rel: This refers to the strategy described above for using SVM-OCC with features generated from RelOCC.
- O-Rnd: This refers to SVM-OCC with features from RND trees (i.e, random features).
- SVM: Instead of using only labeled examples, we use all the examples for training a standard SVM on the propositionalized data. We use *S-Rel* and *S-Rnd* to denote the SVM learned from RelOCC and RND trees respectively.
- RPT: As we are learning conditional discriminative models, we learn relational probability trees (Neville et al. 2003) for the target class as a relational baseline.

We use area under the Precision-Recall curve (AUC-PR) and accuracy to compare our approaches. These datasets have extremely skewed distributions, so using all the examples for testing can achieve a very high precision when predicting the most common label. We use a subset of examples with a reduced skew (twice the negatives to positives) for testing to avoid this issue. The same subset is used for all approaches within each domain. Since one-class SVM methods return only binary predictions we compare the accuracies, while we compare the AUC-PR values for the relational approaches.

We aim to explicitly evaluate the following questions: (Q1) Does RelOCC compare favorably to the baseline relational learning methods for the OCC tasks? and (Q2) Do the features extracted out of RelOCC help the propositional one-class classifier (O-Rel)?

4.1 UW-CSE

UW-CSE dataset (Richardson and Domingos 2006) is a standard SRL dataset developed at University of Washington. The goal is to predict the `advisedBy` relationship between a student and a professor. The data set contains information about professors, students and courses from five different sub-areas of computer science. It includes relations such as `professor`, `student`, `publication`, `advisedBy`, `hasPosition` etc. Since the primary motivation of our approach is one-class classification where all the positive examples are not marked, we use only a subset of positive examples and assume they were the only ones marked by annotators. We pick 20%, 40% and 60% of the labeled examples for training and use all the positives for testing. The results for five-fold cross-validation are shown in Table 2 and Table 3.

% marked	RelOCC	RND	RPT
20%	**0.93**	0.87	0.76
40%	**0.93**	0.81	0.81
60%	**0.92**	0.83	0.88

Table 2: AUC-PR on UW dataset. Bold shows the best approach.

% marked	RelOCC	O-Rel	O-Rnd	S-Rel	S-Rnd
20%	**89.75**	77.39	75.24	60.56	60.58
40%	**89.97**	75.55	74.94	60.56	60.56
60%	**89.60**	76.02	74.43	60.56	60.56

Table 3: Accuracy on UW dataset

As can be seen from the AUC-PR values, our approach outperforms both OCC with random feature selection and discriminative learning using RPTs thus answering Q1 affirmatively. Our approach also has a higher accuracy than both using SVM-OCC and standard SVMs on the propositionalized dataset. The training dataset is highly skewed and

[4] after expanding R^2

as a result basic SVM approaches (S-Rel and S-Rnd) predict the most common label. Thus Q2 can also be answered affirmatively in that using our method for feature selection helps SVM over the other feature selection methods. For all other datasets, S-Rel and S-Rnd show similar behavior where they predict all examples belonging to the negative class and hence we do not show them in other data sets.

4.2 IMDB

IMDB is another popular SRL dataset (Mihalkova and Mooney 2007) containing information about actors, directors and movies obtained from imdb.com. We focused on the task of predicting the `workedUnder` relation in this dataset. We performed five-fold cross-validation and the results are presented in Table 4 and 5. The results are similar to UW-CSE where RelOCC outperforms the SVM-based approaches. But for this dataset, RPT are also able to learn with few positive examples and have similar performance to RelOCC. Both these approaches still outperform RND.

% marked	RelOCC	RND	RPT
20%	**0.98**	0.80	**0.99**
40%	**0.98**	0.88	**0.97**
60%	**0.97**	0.91	**0.97**

Table 4: AUC-PR on IMDB dataset.

% marked	RelOCC	RND	O-Rel	O-Rnd
20%	**93.88**	56.76	81.15	84.10
40%	**94.68**	60.49	81.27	78.90
60%	**94.36**	62.14	78.69	77.53

Table 5: Accuracy on IMDB dataset

4.3 NFL

One of the primary motivations of our work was to learn models for information extraction data sets with only positive labels. To this end, we evaluate our approach on the task of extracting winners of National Football League (NFL) games from sports articles. The annotations only provided few positive examples. We use the Stanford NLP toolkit [5] to convert the text into NLP structures such as parse trees and dependency graphs. Since this dataset only has few annotated positives, we do not drop any labeled examples in our experiments. The results for ten-fold cross-validation on this task are shown in Table 6 and 7.

RelOCC outperforms the other relational approaches thus answering Q1. But as compared to the SVM-based approaches, using the RelOCC features for SVM-OCC (O-Rel) does as well as the relational OCC approach in terms of the accuracy. This shows clearly that our proposed approach serves as a good feature selection method as well (answering Q2). Since this dataset has a large number of possible features, random feature selection is not able to pick the relevant attributes and both RND and O-Rnd are much worse.

[5]http://nlp.stanford.edu/software/corenlp.shtml

RelOCC	RND	RPT
0.63	0.35	0.59

Table 6: AUC-PR (NFL)

RelOCC	RND	O-Rel	O-Rnd
69.95	34.92	**70.22**	47.70

Table 7: Accuracy results (NFL)

4.4 Heart

To evaluate the quality of feature selection, we use the Heart dataset [6] which is a multivariate data set with 13 attributes. The task is to predict the presence of heart disease in patients. We performed four-fold cross-validation and show the results in Tables 8 and 9.

% marked	RelOCC	RND	RPT
20%	**0.47**	0.38	0.39
40%	**0.44**	0.37	0.36
60%	**0.43**	0.38	**0.43**

Table 8: AUC-PR on Heart dataset

While RelOCC is better than both RND and RPT, using SVM-OCC with RelOCC as a feature selection procedure does much better than using the RelOCC model. Since this dataset is originally a propositional dataset, the complex interactions captured by the distance function in RelOCC are not required and can possibly overfit the data. Hence with fewer examples marked (20%), RelOCC does as well as SVM-OCC and better than all the other approaches. The results here show that SVM benefits from the feature selection as it is better than both O-Rnd and the baseline SVM methods answering Q2 affirmatively.

% marked	RelOCC	RND	O-Rel	O-Rnd
20%	**63.85**	38.71	62.73	50.64
40%	50.34	37.67	**62.59**	49.50
60%	46.38	39.31	**60.55**	51.53

Table 9: Accuracy on Heart dataset

5 Conclusion

We presented a non-parametric approach for one-class classification from relational data. We defined a new distance metric based of first-order decision forest and a density estimation model using the distance metric. We can efficiently update the distance metric to improve the classifier's performance. Our approach as a side effect introduces novel and relevant features which can also be used to augment propositional OCC methods. The next step is to apply our approach to other information extraction tasks such as TempEval (UzZaman et al. 2012) and BioNLP (Kim et al. 2009). Integrating other kernels and combination functions into our method might help in further improving the performance.

[6]http://archive.ics.uci.edu/ml/datasets/Heart+Disease

6 Acknowledgments

The authors 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, ARO, or the US government.

References

Anderson, G., and Pfahringer, B. 2008. Exploiting propositionalization based on random relational rules for semi-supervised learning. In *PAKDD*, 494–502.

Bishop, C. 2006. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag New York, Inc.

Blockeel, H., and Raedt, L. D. 1998. Top-down induction of first-order logical decision trees. *Artificial Intelligence* 101:285–297.

Chan, P., and Stolfo, S. 1998. Toward scalable learning with non-uniform class and cost distributions: A case study in credit card fraud detection. In *KDD*.

Chandola, V.; Banerjee, A.; and Kumar, V. 2009. Anomaly detection: A survey. *ACM Computing Surveys* 41(3):15:1–15:58.

Cussens, J. 1998. Using prior probabilities and density estimation for relational classification. In *ILP*.

D'Amato, C.; Staab, S.; and Fanizzi, N. 2008. On the influence of description logics ontologies on conceptual similarity. In *EKAW*.

de Ridder, D.; Tax, D.; and Duin, R. 1998. An experimental comparison of one-class classification methods. In *Conference of the Advanced School for Computing and Imaging, Delft*.

DeComité, F.; Denis, F.; Gilleron, R.; and Letouzey, F. 1999. Positive and unlabeled examples help learning. In *ALT*.

Getoor, L., and Taskar, B., eds. 2007. *Introduction to Statistical Relational Learning*. MIT Press.

Horváth, T.; Wrobel, S.; and Bohnebeck, U. 2001. Relational instance-based learning with lists and terms. *Machine Learning* 43:53–80.

Jensen, D., and Neville, J. 2002. Linkage and autocorrelation cause feature selection bias in relational learning. In *ICML*.

Khan, S., and Madden, M. G. 2009. A survey of recent trends in one class classification. In *Irish conference on Artificial Intelligence and Cognitive Science*.

Kim, J.; Ohta, T.; Pyysalo, S.; Kano, Y.; and Tsujii, J. 2009. Overview of BioNLP'09 shared task on event extraction. In *BioNLP*.

Landwehr, N.; Passerini, A.; De Raedt, L.; and Frasconi, P. 2010. Fast learning of relational kernels. *Machine Learning* 78:305–342.

Liu, F.; Ting, K.; and Zhou, Z. 2012. Isolation-based anomaly detection. *TKDD* 6(1):3.

Mihalkova, L., and Mooney, R. 2007. Bottom-up learning of Markov logic network structure. In *ICML*.

Moya, M., and Hush, D. 1996. Network constraints and multi-objective optimization for one-class classification. *Neural Networks* 9(3):463–474.

Muggleton, S. 1997. Learning from positive data. In *ILP*.

Natarajan, S.; Tadepalli, P.; Dietterich, T.; and Fern, A. 2008. Learning first-order probabilistic models with combining rules. *Annals of Mathematics and AI* 54(1-3):223–256.

Neville, J.; Jensen, D.; Friedland, L.; and Hay, M. 2003. Learning relational probability trees. In *KDD*.

Richardson, M., and Domingos, P. 2006. Markov logic networks. *Machine Learning* 62:107–136.

Schölkopf, B.; Williamson, R.; Smola, A.; Shawe-Taylor, J.; and Platt, J. 2000. Support vector method for novelty detection. In *NIPS*.

Settles, B. 2012. *Active Learning*. Synthesis Lectures on Artificial Intelligence and Machine Learning. Morgan & Claypool.

Tax, D., and Duin, R. 1999. Support vector domain description. *Pattern Recognition Letters* 20:1191–1199.

UzZaman, N.; Llorens, H.; Allen, J.; Derczynski, L.; Verhagen, M.; and Pustejovsky, J. 2012. Tempeval-3: Evaluating events, time expressions, and temporal relations. *CoRR* abs/1206.5333.

Zhu, X., and Goldberg, A. 2009. *Introduction to Semi-Supervised Learning*. Synthesis Lectures on Artificial Intelligence and Machine Learning. Morgan & Claypool Publishers.