Positive and Unlabeled Relational Classification through Label Frequency Estimation

✉ Jessa Bekker[0000-0002-6647-7560] and Jesse Davis[0000-0002-3748-9263]

Computer Science Department, KU Leuven, Belgium
firstname.lastname@cs.kuleuven.be

Abstract. Many applications, such as knowledge base completion and automated diagnosis of patients, only have access to positive examples but lack negative examples which are required by standard relational learning techniques and suffer under the closed-world assumption. The corresponding propositional problem is known as Positive and Unlabeled (PU) learning. In this field, it is known that using the label frequency (the fraction of true positive examples that are labeled) makes learning easier. This notion has not been explored yet in the relational domain. The goal of this work is twofold: 1) to explore if using the label frequency would also be useful when working with relational data and 2) to propose a method for estimating the label frequency from relational positive and unlabeled data. Our experiments confirm the usefulness of knowing the label frequency and of our estimate.

1 Introduction

Relational classification traditionally requires positive and negative examples to learn a theory. However, in many applications, it is only possible to acquire positive examples. A common solution to this issue is to make the closed-world assumption and assume that unlabeled examples belong to the negative class. In reality, this assumption is often incorrect, for example: knowledge bases are incomplete [1], diabetics often go undiagnosed [2] and people do not bookmark all interesting pages. Considering unlabeled cases as negative is therefore sub-optimal. To cope with this, several score functions have been proposed that use only positive examples [3–5].

In propositional settings, having training data with only positive and unlabeled examples is known as Positive and Unlabeled (PU) learning. It has been noted that if the class prior is known, then learning in this setting is greatly simplified. Specifically, knowing the class prior allows calculating the label frequency, which is the probability of a positive example being labeled. The label frequency is crucial as it enables converting standard score functions into PU score functions that can incorporate information about the unlabeled data [6]. Following this insight, several methods have been proposed to estimate the label frequency from PU data [7–10]. To the best of our knowledge, this notion has not been exploited in relational settings. We propose a method to estimate the

label frequency from relational PU data and a way to use this frequency when learning a relational classifier.

Our main contributions are: 1) Investigating the helpfulness of the label frequency in relational positive and unlabeled learning by extending two common relational classifiers to incorporate the label frequency; 2) Modifying TI*c*E, a method for estimating the label frequency in PU data, to operate with relational data, and 3) Evaluating our approach experimentally.

The paper is organized as follows: Section 2 gives an overview of related work. A general background on PU learning and how the label frequency simplifies this problem is presented in Sect. 3. Section 4 discusses how the label frequency can be estimated from PU data. The helpfulness of the label frequency and our method to estimate it from the data are empirically evaluated in Sect. 5. Finally, we conclude in Sect. 6.

2 Related Work

Positive and Unlabeled Learning has been studied mostly in propositional contexts. The proposed methods fall into four categories of approaches. The first and most straightforward approach is to use standard machine learning techniques and just assuming all the unlabeled examples to be negative. The second approach is to look for examples that are very different from the labeled ones and label these as negative. Subsequently, semi-supervised learning methods can be applied [11–15]. The third approach is to formulate an evaluation metric that has only positive or positive and unlabeled data as input and use this for tuning class weights or regularization settings [16–19]. The fourth approach is the approach considered in this paper. It explicitly uses the label frequency to modify traditional classification algorithms [20–23, 6].

In order to use the label frequency, it needs to be given as input. Note that the label frequency can be calculated from the class prior, because the proportion of labeled data is directly proportional to the class prior with the label frequency as the proportionality constant. The class prior could be known from domain knowledge or it could be estimated from a smaller fully labeled dataset. Because this knowledge or data is often not available, several methods have been proposed to estimate it directly from the positive and unlabeled data [6, 24, 7–10].

A few relational positive and unlabeled learning methods exist. The method proposed by Muggleton follows the third positive and unlabeled learning approach and searches for the smallest hypothesis which covers all the positive examples [3]. If the underlying concept is complicated, this is likely to overgeneralize. RelOCC is a positive and unlabeled classification method that incrementally learns a tree-based distance measure which measures the distance to the positive class [25]. The SHERLOCK system is also related because it learns rules from positive examples by evaluating the rules on their statistical relevance and significance, however, it does not utilize the unlabeled data [5].

Knowledge base completion is inherently a positive and unlabeled problem: all the examples that are already in the knowledge base are positive and all

Table 1. Description of terminology used in the paper.

Term	Description
y	Indicator variable for an example to be positive
s	Indicator variable for an example to be labeled
c	Label frequency $\Pr(s = 1 \mid y = 1)$
$\hat{c}$	Estimate of the label frequency
P	(Estimated) number of positive examples
N	(Estimated) number of negative examples
L	Number of labeled examples
U	Number of unlabeled examples
T	Total number of examples

the additional facts that could be included are unlabeled [26]. However, many methods make a closed world assumption when learning models from the original knowledge base and assume everything that is not present to be false [27–30]. Recently, a new score function for evaluating knowledge base completion rules was proposed [1]. It approximates the true precision of a rule by assuming that the coverage of a rule is equal for labeled and unlabeled positives and by estimating the functionality of the relation.

3 PU Learning and the Label Frequency

This section gives some background on PU learning and how the label frequency is used to simplify learning. The methods for using the label frequency are then applied to popular relational classifiers. Table 1 presents the terminology used throughout the paper.

3.1 Positive Unlabeled (PU) Learning

In traditional binary classification, learners are supplied with two types of examples: positive and negative ones. When learning from positive and unlabeled data, commonly referred to as PU learning, there are also two types of examples: positively labeled and unlabeled ones, where the latter can be either positive or negative.

In PU Learning, the labeled positive examples are commonly assumed to be 'selected completely at random' [20, 31, 21, 22, 6]. This means that the probability $c = \Pr(s = 1 \mid y = 1)$ for a positive example to be labeled is constant, which is the same for every positive example. The constant c is called the *label frequency*. Implicit techniques to employ the 'selected completely at random' property are to give more weight to the positive class or to model more noise in the negative class [16–18]. It can also be used explicitly by taking the label frequency into account when training a model [20, 31, 22, 6], which greatly simplifies learning because traditional classifiers can be adjusted to incorporate it in a straightforward manner. This is well-established knowledge for propositional PU learning,

however, to the best of our knowledge, using the label frequency has not been investigated yet for relational PU learning. We briefly review some of the propositional methods and discuss how they can be adjusted to the relational setting.

3.2 Using the Label Frequency to Simplify PU Learning

Elkan and Noto propose to use the label frequency directly to modify traditional classifiers for PU learning [6]. Concretely, they proposed the following two methods:

1. **Probabilistic classifier modification:** This method trains a probabilistic classifier to predict the probability for instances to be labeled. To this end, during training, it considers unlabeled examples as negative. The label frequency is then employed to modify the output probabilities. It transforms the probability that an instance is labeled $\Pr(s = 1|x)$ into the probability that an instance is positive: $\Pr(y = 1|x) = \frac{1}{c}\Pr(s = 1|x)$ [22]. This modified classifier can be used directly or to transform the PU dataset into a probabilistically weighted PN dataset.
2. **Score function modification:** Learning algorithms that make decisions based on counts of positive and negative examples data subsets i can be modified to use counts of labeled and unlabeled examples. The positive and negative counts P_i and N_i can be obtained with $P_i = L_i/c$ and $N_i = T_i - P_I$. Decision trees, for example, assign classes to leaves and score splits based on the positive/negative counts in the potential subsets and can, therefore, be transformed to PU learners [23].

In this paper, we demonstrate how to use these two methods in the relational domain. The proposed solutions c-adjusted TILDE and c-adjusted Aleph are described below.

Relational Probabilistic Classifier Modification: *c*-adjusted TILDE
The first method for using the label frequency requires a probabilistic classifier which predicts the probability that an instance is labeled. The first-order logical decision tree learner TILDE can easily be made probabilistic. Doing so simply requires counting for each leaf i the number of labeled L_i and unlabeled examples U_i that reach i setting the leaf's probability to $\frac{L_i}{U_i+L_i}$ [32]. The tree that predicts the probability for instances to be positive has the same structure as the tree for distinguishing between labeled and unlabeled example, but requires altering the probability in each of its leaves. The new probability in each leaf i is $\frac{1}{c}\frac{L_i}{U_i+L_i}$, where L_i and U_i are defined as above.

Relational Score Function Modification: *c*-adjusted Aleph
The second method for using the label frequency requires a classifier that makes decisions based on the counts N_i and P_i in a subset of the data i. TILDE satisfies this criterion, and so does the rule learner Aleph [33]. The default evaluation

function of Aleph is coverage, which is defined as $P_i - N_i$, where i is the subset of examples that satisfy the rule. To modify Aleph to use the label frequency c, the coverage for each rule r should be calculated as follows:

$$\text{PU coverage} = P_i - N_i = 2P_i - T_i = 2\frac{L_i}{c} - T_i \tag{1}$$

where L_i is the number of labeled (i.e., positive) examples covered by the rule and T_i is the total number of examples covered by the rule.

4 Label Frequency Estimation

To estimate the label frequency in relational PU data, we will use the insights of a propositional label frequency estimator. We first review the original method and then propose a relational version.

4.1 Label Frequency Estimation in Propositional PU data

The propositional estimator is TI*c*E [10]. It is based on two main insights: 1) a subset of the data naturally provides a lower bound on the label frequency, and 2) the lower bound of a large enough positive subset approximates the real label frequency. TIcE uses decision tree induction to find likely positive subsets and estimates the label frequency by taking the maximum of the lower bounds implied by all the subsets in the tree.

The label frequency is the same in subsets of the data because of the 'selected completely at random' assumption, therefore it can be estimated in a subset of the data. Clearly, the true number of positive examples P_i in a subset i cannot exceed the total number of examples in that subset T_i. This naively implies a lower bound: $c = L_i/P_i \geq L_i/T_i$. To take stochasticity into account, this bound is corrected with confidence $1-\delta$ using the one-sided Chebyshev inequality which introduces an error term based on the subset size:

$$\Pr\left(c \leq \frac{L_i}{T_i} - \frac{1}{2}\sqrt{\frac{1-\delta}{\delta T_i}}\right) \leq \delta \tag{2}$$

The higher the ratio of positive examples in the subset, the closer the bound gets to the actual label frequency. The ratio of positive examples is unknown, but directly proportional to the ratio of labeled examples. Therefore, TIcE aims to find subsets of the data with a high proportion of labeled examples using decision tree induction. To avoid overfitting, i.e. finding subsets i where $L_i/P_i > c$, k folds are used to induce the tree and estimate the label frequency on different datasets.

The parameter δ is set such that at least one tenth of the data or 1000 examples are needed to estimate the label frequency with an error term of 0.1: $1/2\sqrt{(1-\delta)/(\delta T_R)} = 0.1$, with $T_R = \min[T/10, 1000]$. This imposed by

$$\delta = \max\left[0.025, \frac{1}{1+0.004T}\right] \tag{3}$$

Table 2. Characteristics of the Datasets

Datasets	#Examples	Class 1 (#)	Class 2 (#)	# Folds
IMDB	268	Actor (236)	Director (32)	5
Mutagenesis	230	Yes (138)	No (92)	5
UW-CSE	278	Student (216)	Professor (62)	5
WebKB	922	Person (590)	Other (332)	4

Table 3. Dataset complexities: The complexities of the models that are trained on the complete and fully labeled datasets. For TILDE, the number of splits in the tree is shown. For Aleph, the number of rules is reported and the average rule length is given in parentheses.

Dataset	TILDE	Aleph Class1	Aleph Class2
IMDB	1	1 (1)	1 (1)
Mutagenesis	6	7 (2.29)	8 (2.25)
UW-CSE	3	5 (1)	5 (1.4)
WebKB	33	32 (3.19)	38 (2.08)

4.2 Label Frequency Estimation in Relational PU Data

We propose TIcER (Tree Induction for c Estimation in Relational data). The main difference with TIcE is that it learns a first-order logical decision tree using TILDE [32]. Each internal node splits on the formula which locally optimizes the gain ratio, considering the unlabeled examples as negative. The examples that satisfy the formula go to the left, the others to the right. Each node in the tree, therefore, specifies a subset of the data, and each subset implies a lower bound on the label frequency through (2). The estimate for the label frequency is the maximal lower bound implied by the subsets:

$$\hat{c} = \max_{i \in \text{subsets}} \left[\frac{L_i}{T_i} - \frac{1}{2} \sqrt{\frac{1-\delta}{\delta T_i}} \right] \tag{4}$$

To prevent overfitting, k folds are used to induce the tree and estimate the label frequency on different datasets. With relational data, extra care should be taken that the data in different folds are not related to each other. The final estimate is the average of the estimates made in the different folds.

5 Experiments

Our goal is to evaluate if knowing the label frequency makes learning from relational PU data easier and if TIcER provides a good estimate of the label frequency. More specifically, we will answer the following questions:

Q1: Does c-adjusted TILDE, the proposed relational probabilistic classifier modification method, improve over classic TILDE when faced with PU data and how sensitive is it to the correctness of $\hat{c}$?

Q2: Does c-adjusted Aleph, the proposed relational score function modification method, improve over classic Aleph when faced with PU data and how sensitive is it to the correctness of $\hat{c}$?
Q3: How well does TIcER estimate the label frequency? In which cases does it perform better or worse?
Q4: How do label frequency adapted methods compare with Muggleton's PosOnly method?

5.1 Datasets

We evaluate our approach on four commonly used datasets for relational classification (Table 2). All datasets are available on Alchemy[1], except for Mutagenesis.[2] The classes of WebKB are disjunctive concepts. Person contains web pages from students, faculty and staff and Other contains web pages from departments, courses, and research projects. To get an intuition of the complexity of the concepts to be learned, Table 3 shows how big the TILDE and Aleph models are if they are trained on the complete dataset with labels for all examples. The datasets were converted to PU datasets by selecting some of the positive examples at random to be labeled. The labeling was done with frequencies $c \in \{0.1, 0.3, 0.5, 0.7, 0.9\}$. Each has five different random labelings.

5.2 Methods

For our experiments we used the following PU classifiers:

- c-adjusted TILDE, as described in Sect. 3.2.
- c-adjusted Aleph, as described in Sect. 3.2.
- Aleph, taking unlabeled examples as negative ($\hat{c} = 1$)
- TILDE, taking unlabeled examples as negative ($\hat{c} = 1$)
- PosOnly: Muggleton's approach, implemented in Aleph [3].[3]

All classifiers, including TILDE when used for TIcER, use standard settings, with the exceptions of requiring PosOnly rules to cover at least two example and allowing infinite noise and exploration in Aleph.

For the c-adjusted methods, an estimate of the label frequency c is required. This estimate $\hat{c}$ can be the correct label frequency c or the estimate obtained by our method TIcER. For the sensitivity experiments, the $\hat{c}$ is varied in $c \pm \Delta$ with $\Delta \in \{0, 0.05, 0.15, 0.25\}$. The naive baseline where unlabeled examples are considered to be negative can be seen as a special case of the c-adjusted methods with $\hat{c} = 1$.

k-fold cross-validation was applied for validation, i.e., $k - 1$ folds were used for learning the classifier and the other fold to evaluate it. TIcER also needs

[1] `http://alchemy.cs.washington.edu/data/`
[2] `http://www.cs.ox.ac.uk/activities/machlearn/mutagenesis.html`
[3] `http://www.cs.ox.ac.uk/activities/machinelearning/Aleph/aleph`

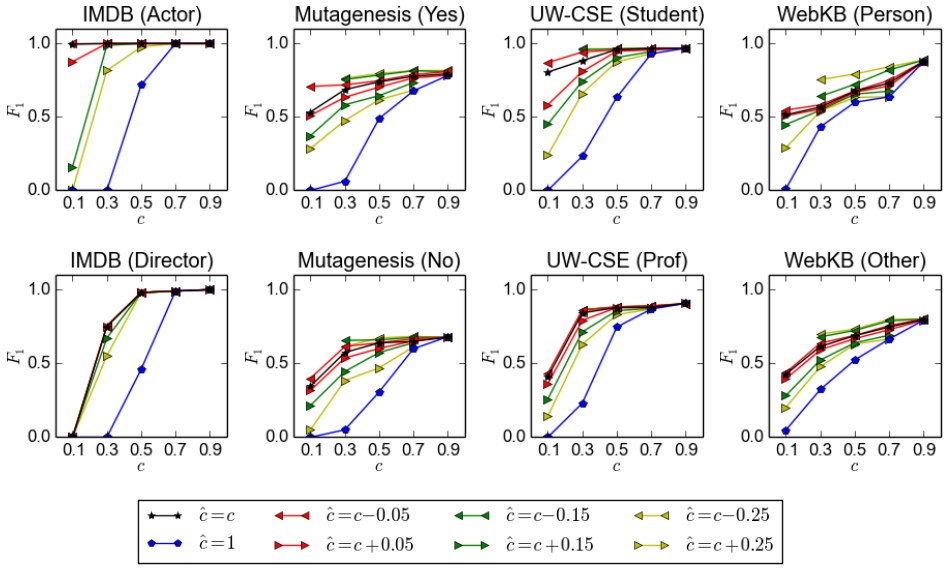

Fig. 1. TILDE sensitivity to c. Taking the label frequency into account clearly improves the classifier. The F_1 does decrease as fewer labeled examples are provided. It is striking that underestimates and overestimates of the label frequency have very different effects on the performance. c-adjusted TILDE is very sensitive to overestimates, but not to underestimates. In fact, in some cases it even benefits from underestimates!

folds for estimation, it used 1 fold for inducing a tree and the other $k-2$ folds for bounding the label frequency.

The classifiers are compared using the F_1 score and the average absolute error of the estimated cs are reported.

5.3 c-adjusted TILDE: Performance and Sensitivity to $\hat{c}$

This section aims to answer **Q1**: Does c-adjusted TILDE, the proposed relational probabilistic classifier modification method, improve over classic TILDE when faced with PU data and how sensitive is it to the correctness of $\hat{c}$? To this end, TILDE was adjusted with different estimates for the label frequency that deviate from the true label frequencies with fixed values Δ. The adjusted versions are compared to the naive method which considers unlabeled examples as negative, i.e., $\hat{c}=1$. The results are presented in Fig. 1.

As expected, taking the label frequency into account improves the classifier. A striking observation is that overestimates of the label frequency c can severely degrade performance, while underestimates may even improve performance. This is because of the modification method: only the leaf probabilities are altered. Therefore, an underestimate makes leaves with at least one labeled example

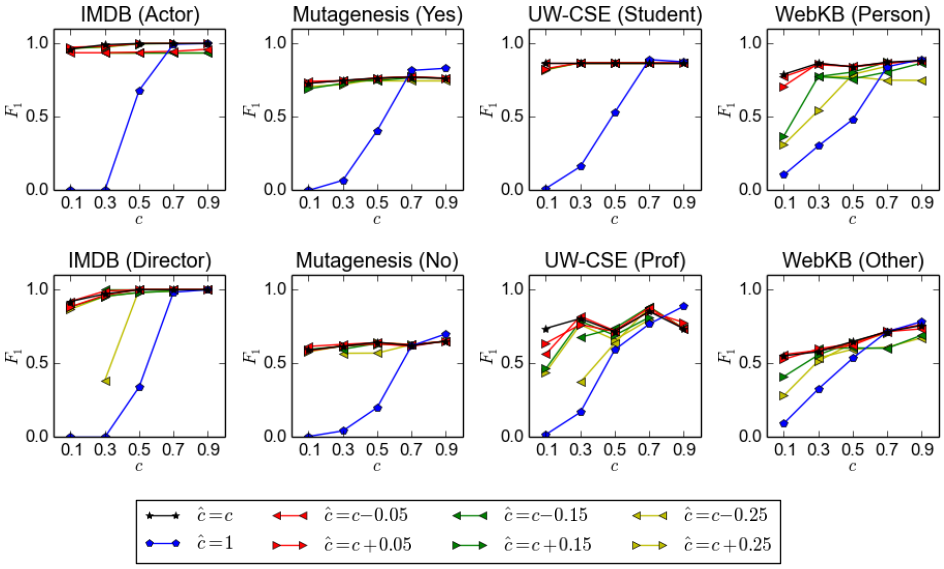

Fig. 2. Aleph sensitivity to c. Considering the label frequency clearly substantially improves the classifier: as the number of labeled examples decreases, the F_1 score barely drops. Aleph is not very sensitive to the label frequency c, except when there are few positive examples to start with (IMDB-director and UW-CSE-Prof) or when the target concept is complex (WebKB). Even in these cases, a bad estimate for the label frequency is better than taking the unlabeled examples as negative ($\hat{c} = 1$).

more likely to classify instances as positive, while leaves without any labeled examples will always classify instances as negative.

5.4 c-adjusted Aleph: Performance and Sensitivity to $\hat{c}$

This section aims to answer **Q2**: Does c-adjusted Aleph, the proposed relational score function modification method, improve over classic Aleph when faced with PU data and how sensitive is it to the correctness of $\hat{c}$? To this end, Aleph was adjusted with different estimates for the label frequency that deviate from the true label frequencies with fixed values Δ. The adjusted versions are compared to the naive method which considers unlabeled examples as negative, i.e., $\hat{c} = 1$. The results are presented in Fig. 2.

Taking the label frequency into account drastically improves the classifier: the F_1 score barely drops when the label frequency decreases. In most cases, a reasonable approximation of the label frequency yields an equivalent performance to using the true label frequency. Two exceptions are 1) when there are few positive examples in the fully labeled dataset, and 2) when the target concept is very complex. But even in these cases, the performance does not suffer that

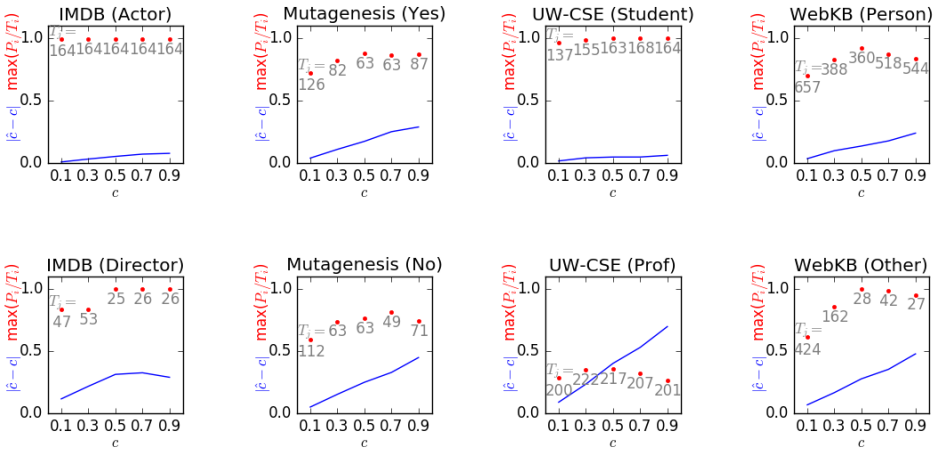

Fig. 3. Label Frequency Estimates. The estimate is expected to be good if a large enough subset with a high proportion of positives was found, this is confirmed by the experiments. For example, the worst results, for UW-CSE (Prof), are explained by the low positive proportions. Subset i is the subset with the maximum purity and subset j is the largest subset that is at least 90% as pure.

much, especially when compared to simply assuming that all unlabeled examples belong to the negative class.

5.5 TI*c*ER Evaluation

This section aims to answer **Q3**: How well does TI*c*ER estimate the label frequency? In which cases does it perform better or worse? To this end, TI*c*ER was used to estimate the label frequency c and compared to the true label frequency in all the training folds of all the datasets. Based on the theory, it is expected that TI*c*ER works well when it can find subsets in the dataset that are purely positive and contain a sufficient number of examples. To check this, the maximal proportion of true positives over all the used subsets was recorded for each setting. We could look at the size of this purest subset to check if a large subset is found. However, the purest subset could be very small and another subset that is almost as pure could be very big. Therefore, we recorded the largest subset that is at least 90% as pure as the purest subset. Figure 3 shows the averaged absolute error $|\hat{c} - c|$, purity $\max(P_i/T_i)$ of the purest subset i and size T_j of the largest subset j with purity close to that of the purest subset, for different label frequencies c. Figures 4 and 5 compare the performance of TILDE and Aleph respectively when adjusted with the TI*c*ER estimate, the true label frequency and without adjusting it.

TI*c*ER gives reasonable results most of the time. The experiments confirm our expectations: it performs worse when it fails to find subsets with a high ratio

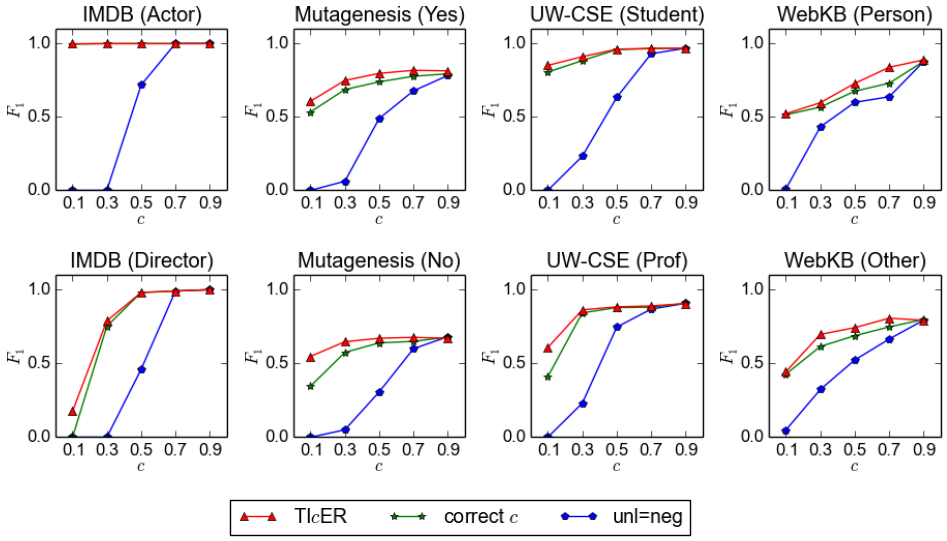

Fig. 4. TIcER-adjusted TILDE. Adjusting TILDE with the TIcER estimate gives very similar results to adjusting it with the true label frequency, sometimes even better. This is explained by TIcER giving underestimates.

of positive examples or when the subsets contain few examples. Although the estimates are not perfect, they still can improve the performance of TILDE and Aleph. Most of the time, the performance using the estimated label frequency is close to the performance of using the true label frequency. TILDE even gives better results with the estimate than with the true label frequency, this is because TIcER estimates the label frequency by looking for the maximum lower bound and hence tends to give underestimates. Aleph performs worse for the cases where it is sensitive to the label frequency. This is notably the case for UW-CSE.

5.6 Method Comparison

This section aims to answer **Q4**: How do label frequency adapted methods compare with Muggleton's PosOnly method? To this end, we compare TIcER-adjusted TILDE and Aleph with PosOnly. The results are presented in Fig. 6.

The label frequency indeed makes learning from PU data easier, as it gives similar or better results than PosOnly. It is especially interesting that for the most complex dataset (WebKB) PosOnly is outperformed. The label frequency-based methods are only outperformed when there are exceptionally few labeled examples or no close-to-pure subsets in the data. Because using the label frequency adjust existing methods, it can benefit from any advancements and optimizations made to traditional classifiers.

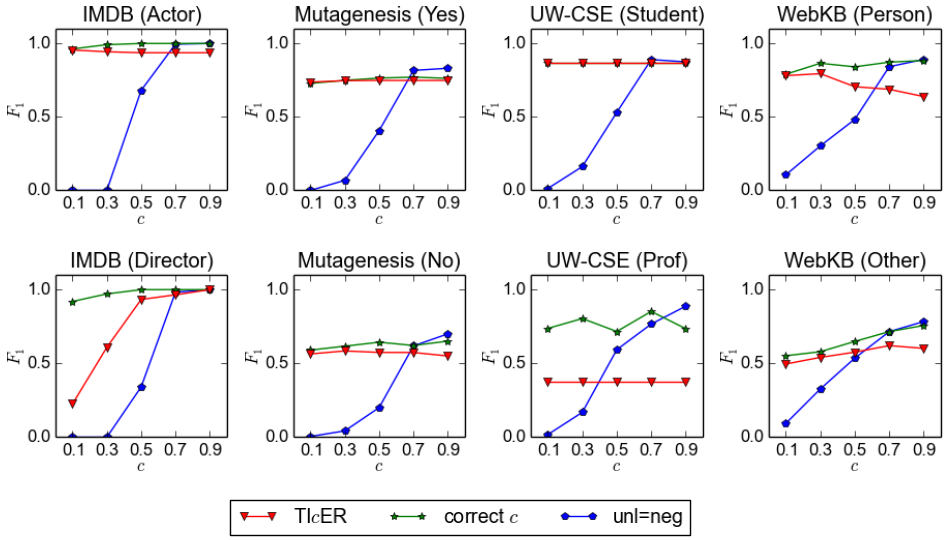

Fig. 5. TIcER-adjusted Aleph. Adjusting Aleph with the TIcER estimate gives for most cases similar results to adjusting it with the true label frequency. It performs worse for the datasets where Aleph is very sensitive to c. UW-CSE-Prof is doubly problematic because it is both sensitive to the label frequency and the most difficult dataset for TIcER.

6 Conclusions

For propositional PU classification tasks, it has long been known that knowing the label frequency greatly simplifies the problem. We transferred this idea to the relational classification tasks and make the same conclusion here. Adjusting established classifiers such as TILDE and Aleph in very simple ways perform equally well as Muggleton's PosOnly method. For the most complex dataset, PosOnly is even outperformed. Because only small adjustments in traditional classifiers are needed, this PU classification method will improve as traditional classifiers improve.

When the label frequency is unknown, it needs to be estimated from the positive and unlabeled data. We propose a TIcER, a relational version of TIcE, which employs decision trees to find pure positive subsets in the data and uses these to estimate the label frequency. This method works well when it can find highly positive subsets of the data that contain enough examples.

Acknowledgements

JB is supported by IWT (SB/141744). JD is partially supported by the KU Leuven Research Fund (C14/17/070, C32/17/036), FWO-Vlaanderen (SBO-150033,

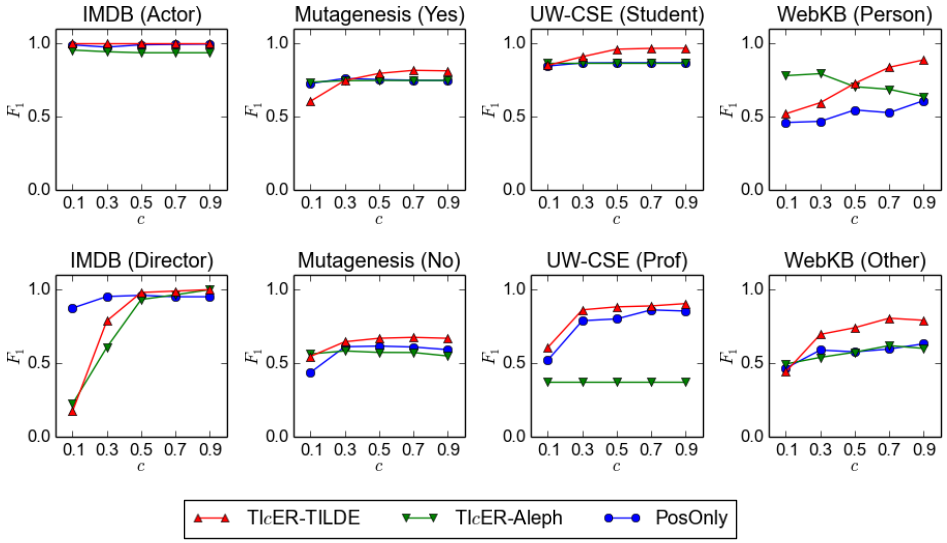

Fig. 6. Comparison of methods. In most cases, the methods give similar results, which supports the claim that using the label frequency simplifies PU learning, also in the relational domain. It is interesting to see that for the most complex dataset (WebKB) Muggleton's PosOnly is outperformed. The only situations where any of the c-adjusted methods perform significantly worse than PosOnly are those with an extremely small number of labeled examples (IMDB-Director with low label frequency) or when the estimate is extremely bad because of the lack of pure positive subsets (UW-CSE)

G066818N, EOS-30992574, T004716N), Chist-Era ReGround project, and EU VA project Nano4Sports.

References

1. Zupanc, K., Davis, J.: Estimating rule quality for knowledge base completion with the relationship between coverage assumption. In: Proceedings of the 27th International Conference on World Wide Web (WWW18). (2018)
2. Claesen, M., De Smet, F., Gillard, P., Mathieu, C., De Moor, B.: Building classifiers to predict the start of glucose-lowering pharmacotherapy using Belgian health expenditure data. arXiv preprint arXiv:1504.07389 (2015)
3. Muggleton, S.: Learning from positive data. In: Selected Papers from the 6th International Workshop on Inductive Logic Programming. (1996) 358–376
4. McCreath, E., Sharma, A.: ILP with noise and fixed example size: A bayesian approach. In: Proceedings of the Fifteenth International Joint Conference on Artifical Intelligence. (1997) 1310–1315
5. Schoenmackers, S., Davis, J., Etzioni, O., Weld, D.S.: Learning first-order Horn clauses from web text. In: Proceedings of Conference on Empirical Methods on Natural Language Processing. (2010)
6. Elkan, C., Noto, K.: Learning classifiers from only positive and unlabeled data. In: Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining. (2008) 213–220
7. du Plessis, M.C., Niu, G., Sugiyama, M.: Class-prior estimation for learning from positive and unlabeled data. Machine Learning (2015) 1–30
8. Jain, S., White, M., Radivojac, P.: Estimating the class prior and posterior from noisy positives and unlabeled data. In: Advances in Neural Information Processing Systems. (2016)
9. Ramaswamy, H.G., Scott, C., Tewari, A.: Mixture proportion estimation via kernel embedding of distributions. In: Proceedings of International Conference on Machine Learning. (2016)
10. Bekker, J., Davis, J.: Estimating the class prior in positive and unlabeled data through decision tree induction. In: "Proceedings of the 32th AAAI Conference on Artificial Intelligence". (2018)
11. Liu, B., Lee, W.S., Yu, P.S., Li, X.: Partially supervised classification of text documents. In: Proceedings of the International Conference on Machine Learning. (2002) 387–394
12. Li, X., Liu, B.: Learning to classify texts using positive and unlabeled data. In: Proceedings of the International Joint Conference on Artifical Intelligence. (2003) 587–592
13. Yu, H.: Single-class classification with mapping convergence. Machine Learning **61**(1-3) (2005) 49–69
14. Li, X.L., Yu, P.S., Liu, B., Ng, S.K.: Positive unlabeled learning for data stream classification. In: Proceedings of the 2009 SIAM International Conference on Data Mining. (2009) 259–270
15. Nguyen, M.N., Li, X.L., Ng, S.K.: Positive unlabeled leaning for time series classification. In: Proceedings of the International Joint Conference on Artificial Intelligence. (2011) 1421–1426
16. Lee, W.S., Liu, B.: Learning with positive and unlabeled examples using weighted logistic regression. In: Proceedings of the International Conference on Machine Learning. Volume 3. (2003) 448–455
17. Liu, Z., Shi, W., Li, D., Qin, Q.: Partially supervised classification–based on weighted unlabeled samples support vector machine. In: International Conference on Advanced Data Mining and Applications. (2005) 118–129

18. Mordelet, F., Vert, J.P.: A bagging SVM to learn from positive and unlabeled examples. Pattern Recognition Letters **37** (2014) 201–209
19. Claesen, M., De Smet, F., Suykens, J.A., De Moor, B.: A robust ensemble approach to learn from positive and unlabeled data using SVM base models. Neurocomputing **160** (2015) 73–84
20. Denis, F.: Pac learning from positive statistical queries. In: International Conference on Algorithmic Learning Theory. (1998) 112–126
21. Liu, B., Dai, Y., Li, X., Lee, W.S., Yu, P.S.: Building text classifiers using positive and unlabeled examples. In: Proceedings of the Third IEEE International Conference On Data Mining. (2003) 179–186
22. Zhang, D., Lee, W.S.: A simple probabilistic approach to learning from positive and unlabeled examples. In: Proceedings of the 5th Annual UK Workshop on Computational Intelligence. (2005) 83–87
23. Denis, F., Gilleron, R., Letouzey, F.: Learning from positive and unlabeled examples. Theoretical Computer Science (2005) 70–83
24. du Plessis, M.C., Sugiyama, M.: Class prior estimation from positive and unlabeled data. IEICE Transactions **97-D** (2014) 1358–1362
25. Khot, T., Natarajan, S., Shavlik, J.W.: Relational one-class classification: A non-parametric approach. In: Proceedings of the 28th AAAI Conference on Artificial Intelligence. (2014)
26. Galárraga, L., Teflioudi, C., Hose, K., Suchanek, F.: Fast rule mining in ontological knowledge bases with amie+. The VLDB Journal (2015) 707–730
27. Lao, N., Subramanya, A., Pereira, F., Cohen, W.W.: Reading the web with learned syntactic-semantic inference rules. In: Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning. (2012) 1017–1026
28. Socher, R., Chen, D., Manning, C.D., Ng, A.: Reasoning with neural tensor networks for knowledge base completion. In: Advances in Neural Information Processing Systems 26. (2013) 926–934
29. Gardner, M., Talukdar, P.P., Krishnamurthy, J., Mitchell, T.M.: Incorporating vector space similarity in random walk inference over knowledge bases. In: EMNLP. (2014)
30. Neelakantan, A., Roth, B., McCallum, A.: Compositional vector space models for knowledge base completion. Proceedings of the Annual Meeting of the Association for Computational Linguistics (2015)
31. De Comité, F., Denis, F., Gilleron, R., Letouzey, F.: Positive and unlabeled examples help learning. In: International Conference on Algorithmic Learning Theory. (1999) 219–230
32. Blockeel, H., De Raedt, L.: Top-down induction of first-order logical decision trees. Artificial intelligence (1998) 285–297
33. Srinivasan, A.: The Aleph manual (2001)