Learning to Extract Relations for Relational Classification

Steffen Rendle, Christine Preisach, and Lars Schmidt-Thieme

Machine Learning Lab, University of Hildesheim, Germany
{srendle,preisach,schmidt-thieme}@ismll.uni-hildesheim.de

Abstract. Relational classifiers use relations between objects to predict the class values. In some cases the relations are explicitly given. In other cases the dataset contains implicit relations, e.g. the relation is hidden inside of noisy attribute values. To apply relational classifiers for this task, the relations have to be extracted. Manually extracting relations by a domain expert is an expensive and time consuming task. In this paper we show how extracting relations in datasets with noisy attribute values can be learned. Our method LRE uses a regression model to learn and predict weighted binary relations. We show that LRE is able to extract both equivalence relations and non-constrained relations. Secondly we show that relational classifiers using relations automatically extracted by LRE achieve comparable classification quality as classifiers using manually labeled relations.

1 Introduction

Relational classifiers use relations between instances to predict class values ('entity classification task'). An example is classifying papers by categories using relations like *CommonAuthor*, *CommonCitations* or *SameConference*, where the relations indicate the number of co-authors, co-citations or the same conference given two papers. For problems where such relations are present, relational classifier have shown to outperform state-of-the-art non-relational classifiers like Support Vector Machines [1–3]. The drawback of relational classifiers is that they can only be applied in scenarios where the relations are explicitly given. In most real-world scenarios this is not the case. If the relations are not directly given, they have to be annotated manually by domain experts. This is a time consuming and expensive task.

In this paper we propose the method LRE ('Learning Relation Extraction') for automatically extracting relations from a noisy dataset. The method is based on learning a binary regression model for pairs of instances. As the method uses training data, it can adapt itself to different tasks and domains. This makes it superior to handcrafted or heuristic extraction rules.

The overall contributions of our paper are as following: (1) We propose the supervised method LRE for extracting relations. These relations can be used in relational classifiers for the task of entity classification. (2) Our evaluation indicates that learning the relation with our LRE method is successful and results in a comparable performance to manually labeled relations.

ID	Author	Paper title	Conference
p_1	Yoav Freund	Boosting a weak learning algorithm by majority.	COLT
p_2	Yoav Freund, H. Sebastian Seung, Eli Shamir, Naftali Tishby	Prediction, and query by committee	NIPS
p_3	S Rosset, E Segal	Boosting density estimation	NIPS

Table 1. Data from the domain of paper topic classification.

2 Related work

Our LRE approach has parallels to the field of record linkage [4], where an equivalence relation should be learned. We share the feature extraction step with work in this field. Here also several heuristic pairwise similarity measures are used to create pairwise features [5, 6]. In other components, our approach differs from the work in record linkage as the properties of the relation to predict differs. In record linkage an equivalence relation has to be learned. Thus typically a probabilistic classifier is learned over pairs [5]. Afterwards clustering is performed to create a binary equivalence relation. This approach has been suggested by Preisach et al. [7] to extract relations for relational learning. In contrast to this we learn a regression model to predict weighted relations.

3 Problem setting

Throughout the paper we use a bibliographic scenario for illustration and evaluation. In this scenario a data set of research papers is given and the category of the paper, e.g. *Machine Learning* or *Artificial Intelligence* should be predicted. We will denote the set of objects – here papers – by $X = \{x_1, \ldots, x_n\}$ and a relation by R. An example of such a data set is given in table 1. Here each paper is described by the attributes *Author*, *Paper title* and *Conference*.

The field values of each attribute can be seen as a relation over a pair of objects – here over two papers. For example the *Conference* attribute values of p_2 and p_3 in table 1 are identical and thus both paper can be linked over a boolean relation SameConference. Similarly, p_1 and p_2 have one author in common and thus CommonAuthors$(p_1, p_2) = 1$. Having such a data set like in figure 1, the relations SameConference and CommonAuthors can easily be constructed. The values of the two binary relations of the given example can be found in table 2. With these two relations any relational classifier can be applied to predict the paper category.

3.1 Relational classification

A probabilistic relational classifier $P(c|x)$ is a function to predict a class value $c \in C$ for an object $x \in X$ given j relations $R_1, \ldots, R_j : X^2 \to T$. Where T is the range of the relation. We will deal with cases of $T \subseteq \mathbb{R}$.

CommonAuthors			
ID	p_1	p_2	p_3
p_1	1	1	0
p_2	1	4	0
p_3	0	0	2

SameConference			
ID	p_1	p_2	p_3
p_1	true	false	false
p_2	false	true	true
p_3	false	true	true

Table 2. Two binary relations over the papers of table 1. CommonAuthors is a weighted relation, SameConference is a boolean relation.

ID	Author	Paper title	Conference
p_1	Freund, Y.	Boosting a weak learning algorithm by majority.	3rd annual workshop on computational learning theory
p_2	Yoav Freund, H. Sebastian Seung, E Shamir, and Naftali Tishby	Prediction, and query by committee	advances in neural information processing systems
p_3	S Rosset, E Segal	Boosting density estimation	NIPS 2003

Table 3. Data sample of table 1 with noise in the Conference and Author attributes.

There are many approaches for relational classifiers. One example is the *probabilistic relational classifier (PRN)* introduced by Macskassy and Provost [2]. It calculates the class probability as the weighted arithmetic mean of the class-membership probabilities of the related objects given a relation R:

$$P(c|x) = \frac{1}{Z} \cdot \sum_{x' \in X} R(x, x') \cdot P(c|x') \tag{1}$$

Where $Z := \sum_{x' \in X} R(x, x')$ is the normalization constant.

Preisach and Schmidt-Thieme [8] have introduced extensions of this algorithm. One extension is not only to take into account direct neighbors but also indirectly linked objects by longer paths. As R is often very sparse, this can be seen as a densification of the relation graph. The model equation of PRN (see (1)) is for a single relation R. PRN can be extended [8] to a multi-relational classifier EPRN by combining several single-relational PRN classifiers by ensembling techniques, e.g. stacking.

3.2 Noisy data sets

Relational classifiers are known to result in high quality predictions. However real world data often does not state the relations explicitly. Instead the data set might look like in table 3. Here for example the identity between '*NIPS 2003*' and '*advances in neural information processing systems*' is not obvious. Thus a relational model cannot be run directly on such noisy data sets.

There are many reasons why relations in data sets are not stated explicitly but are hidden. One obvious reason are errors that occur when humans work with an information system, e.g. typos. Secondly, often several abbreviations can be used, e.g. NIPS stands for ***N**eural **I**nformation **P**rocessing **S**ystems*. When writing names the first name can be placed after the family name – e.g. *John Smith* is *Smith, John*. Other places for noise in textual encoded relations are ambiguities – e.g. there are two different people with the same name.

This paper deals with the problem of extracting relations from such noisy data sets. We use an adaptive, supervised method that learns to extract relations given some training data.

3.3 Type of relations

There are several different types of relations that can be extracted. They can be distinguished by the range T of their target. Important ranges are:

- **Boolean relations** where $T = \{0, 1\}$, e.g SameConference or SamePublisher.
- **Probabilistic relations** where the target is a probability score and thus the range is $T = [0, 1]$.
- **Weighted relations** where a real-valued score is attached to each pair: $T \subseteq \mathbb{R}$, e.g. CommonAuthors.

Our learning method for relations will handle the most general case of weighted relations.

4 Learning weighted relations

In the following we describe our method LRE for **L**earning **R**elation **E**xtraction from a noisy data set like in table 3. Our proposed method extracts pairwise features given two objects. Based on these features, a regression model for R is learned and applied to predict $\hat{R}$. The overall algorithm can be found in algorithm 1.

Next we describe the main components of LRE in detail, i.e. extracting pairwise features, learning the regression model and generating candidates for scaling to large data sets.

4.1 Extracting pairwise features

The overall goal of our method is to extract relations from noisy attribute values of a data set. Let $a_1, \ldots, a_k : X \to V$ be the noisy attributes of our dataset. E.g. a_1 might return the noisy list of the authors of a specific paper, a_2 the conference string, etc.

As the target relation is defined over a pair of objects, feature extraction aims to create meaningful pairwise information from the attributes. Thus feature

Algorithm 1 LRE: Learning Relation Extraction

procedure LEARNRELATION(X, X_{tr}, R_{tr})
outputs a relation $\hat{R}$ for X

$\forall (x,y) \in \left(X^2 \cup X_{\text{tr}}^2\right) : f(x,y) \leftarrow (f_1(x,y), \ldots, f_l(x,y))$ ▷ extract features
Training:
$C \leftarrow$ GENERATECANDIDATES(X_{tr}) ▷ generate candidate pairs
$D_{\text{tr}} \leftarrow \{(f(x,y), R_{\text{tr}}(x,y)) | (x,y) \in C\}$ ▷ create training-set
TRAIN(D_{tr}) ▷ training the regression model
Prediction:
$\forall (x,y) \in X^2 : \hat{R}(x,y) \leftarrow$ PREDICT($f(x,y)$)

return $\hat{R}$
end procedure

extraction can be formulated as a function that generates a real valued feature vector for a pair of objects:

$$f : X^2 \rightarrow \mathbb{R}^l \tag{2}$$

To create a real-valued vector of length l from a pair of objects, several heuristic similarity measures $f_1, \ldots, f_l$ can be applied. Examples of well-known similarity/ distance functions for strings are TFIDF cosine similarity, Overlap index or Levenshtein distance. Details about these and many more similarity measures can be found in [9]. Table 4 shows an example for feature extraction for the COMMONAUTHORS relation.

4.2 Relation learning

Based on the pairwise features, a regression model is learned to estimate the relation weight between a pair of objects. The model is trained on a labeled subset of objects X_{tr}, where the true relation weight $R(x,y)$ between object pairs $(x,y) \in X_{\text{tr}}^2$ is known. The features of a pair of objects for training the model and later on to predict the unknown weights consist of the elements of the feature vector $f = (f_1, \ldots, f_l)$.

In our implementation we applied a Support Vector Regression (SVR) model using the implementation of libSVM [10]. In general, any other regression method could also be used, e.g. linear regression or ridge regression.

4.3 Scaling

The method proposed so far has a runtime that is quadratic $O(n^2)$ in the number of objects X because the binary relation R has $|X^2|$ possible entries. For large data sets this method is not applicable any more. A solution to the problem relies on the fact that most relations are very sparse, i.e. have many 0 entries. We suggest to use blocking techniques from the field of record linkage to handle

Pair	Overlap(Author)	TFIDF-Sim(Author)	Target Value
(p_1, p_2)	0.5	0.2	1
(p_1, p_3)	0.0	0.0	0
(p_2, p_3)	0.25	0.03	0

Table 4. Features for learning the relation CommonAuthors from the noisy data of table 3 by using two heuristic pairwise similarity functions.

such cases. Blocking is a method to reduce the number of pairs, by discarding pairs that are obviously 0. For relations that are equivalence relations any of the proposed blocking techniques from the field of record linkage can be used for GenerateCandidates (see algorithm 1). An overview about blocking for record linkage is given by Baxter et al. [11]. For scaling to the problem of predicting equivalence classes that have both big and many classes, the method proposed by Rendle and Schmidt-Thieme [12] can be applied.

In the general case of predicting an arbitrary weighted relation, we suggest to generate candidates by considering only pairs of objects whose similarity based on a cheap measure exceeds a certain threshold.

5 Evaluation

In our evaluation we investigate if our method successfully learns relations for relational classification. We measure the success in two tasks:

1. **Relation extraction task**: This task measures how good the extracted relation $\hat{R}$ approximates the true (manually) labeled relation R in terms of RMSE over a set S of pairs:

$$E_{\text{RMSE}}(S) = \sqrt{\frac{1}{|S|} \sum_{(x,y) \in S} \left(R(x,y) - \hat{R}(x,y) \right)^2} \tag{3}$$

2. **Entity classification task**: Here the quality of a relation R is measured in terms of how a relational classifier c performs by using the relation. As quality measure we use the classification accuracy:

$$Q^R_{\text{Acc}}(X) = \frac{|\{x \in X : c_R(x) = y_x\}|}{|X|} \tag{4}$$

   Where y_x is the true class label for x and $c_R(x)$ is the prediction of the relational classifier for x using the relation R. We use this measure to compare if a learned relation $\hat{R}$ leads to comparable results as the manually labeled relation R. This means we compare $Q^{\hat{R}}_{\text{Acc}}(X)$ to $Q^R_{\text{Acc}}(X)$.

In all, the entity classification task is the more important task and the goal is to achieve a classification quality with a learned relation as high as with a manually labeled one.

5.1 Dataset

We perform our evaluation on the relational Cora dataset provided by McCallum[1]. This dataset contains papers with their category, author string, citations, conference and journal string. We use a subset with the 12 subcategories of 'Artificial Intelligence', i.e. 'Machine Learning', 'Knowledge Representation', 'Data Mining', etc. We removed papers without authors, citations and neither conference nor journal. In total this subset has 3298 papers.

In the Cora dataset, for the author and citation relation the true relation is already present. For authors the noisy author string is also present. Given this string we try to learn the true author relation. Secondly, we merged the noisy conference and journal attribute. For this merged attribute, we manually labeled a new relation SameConference which states if two papers were published in the same conference proceedings or were published in the same journal. This relation is an equivalence relation. Again in our experiments we try to learn this relation from the noisy string.

5.2 Model setup

We choose a support vector regression as regression model. The hyperparameters are optimized in each repetition via grid search using cross-validation. As similarity functions we use TFIDF, relative overlap in tokens and absolute overlap in tokens. We use different variants of tokenizers that extract words, 2-grams and 3-grams. Each similarity function is run with each tokenizer. For speedup we use a canopy blocker for the equivalence relation SameConference; for CommonAuthor we use the blocking method described in section 4.3. For relational classification we use an ensemble of PRN models. The ensemble is created by stacking with an SVM as meta classifier [8].

5.3 Learning relations

In the first experiment we split the dataset X in two disjoint parts of equal size X_{train} and X_{test} with $X_{\text{train}} \cup X_{\text{test}} = X$. We assume that the relations between elements in X_{train} are known – i.e. the relation weights correspond to the manually labeled weights. We then try to predict with LRE all other relation weights, i.e. $\hat{R}$ on $(X \setminus X_{\text{train}})^2$. The results of this relation extraction task is shown in figure 1. We compare the error of LRE to a constant baseline that predicts constantly a weight of 0. As the relations are very sparse – i.e. the weight of most relation pairs is 0 – the constant method results in a low RMSE error. Compared to the constant method the relation extracted by LRE results in an even lower RMSE. Eventhough the relation predicted by the constant method has a low RMSE, this relation is useless for the entity classification task as no objects are linked. With such a relation the relational classifier can only predict the majority class as the relation contains no information. In contrast to this the

[1] http://www.cs.umass.edu/∼mccallum/data/cora-classify.tar.gz

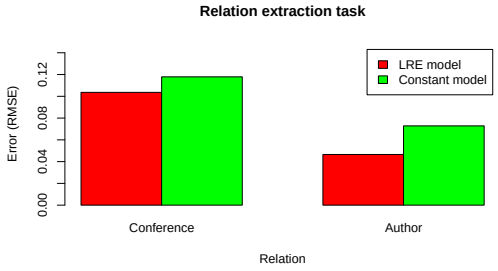

Fig. 1. Error (RMSE) between the true and the predicted relation.

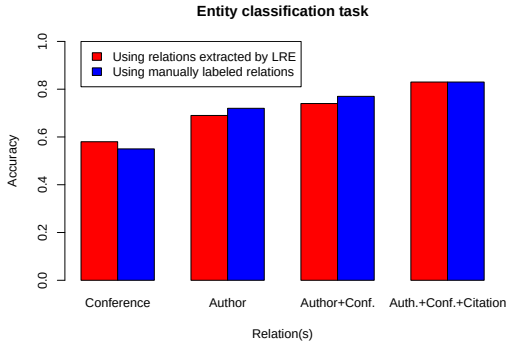

Fig. 2. Comparing the quality of the entity classification task for manually labeled and automatically extracted relations. The goal is to achieve with extracted relations a classification score that is as good as with manually labeled ones.

relations extracted by LRE are also successful in solving the entity classification task.

For the entity classification task the entity labels on X_{train} are assumed to be known whereas the labels on X_{test} are unknown and quality is measured on X_{test}. Figure 2 shows the classification accuracy using several combinations of relations. For every combination we report the accuracy with manually labeled relations and the relations automatically extracted by LRE. The citation relation is always manually labeled. As you can see, the classification quality using the learned relations is very similar to the results using manually labeled relations. For the case of the SameConference relation, prediction quality is even slightly better with the learned relation. For this relation the errors that the relation extraction makes seem to be informative for solving the entity classification task.

All results were repeated 10 times with different train/test splits. We report the means; the standard deviation within each experiment is below 0.01 accuracy.

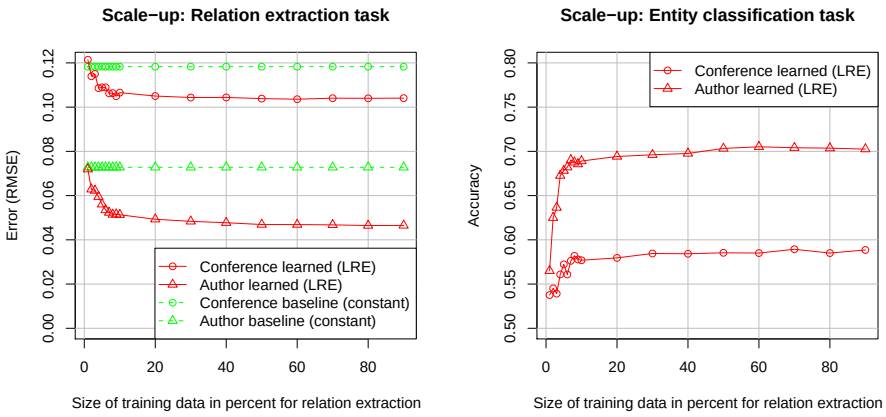

Fig. 3. Quality of the relation extraction task and error (RMSE) of the entity classification using learned relations with increasing amount of labeled data for learning the relation.

5.4 Amount of training data

In the second experiment we vary the amount of training data for learning the relation. This should show how much pairs of a relation have to be labeled before this relation can be learned successfully and applied to the entity classification task. Our protocol is as following: From the dataset X a test set X_{test} of one third is put aside. Within the remaining 66% we take an increasing proportion $X^{\text{rel}}_{\text{train}}$ from 1 to 90 percent. We assume that the relation weights within $X^{\text{rel}}_{\text{train}}$ are known. We than predict the whole relation on X and evaluate it on X_{test} (see figure 3). We repeated the experiment 8 times with different training data.

Then we use the predicted relation for the task of entity classification. Here we assume that the entity labels on $X \setminus X_{\text{test}}$ are known. We evaluate the quality of the entity classification task on X_{test}. With this evaluation protocol we can measure how many proportions of a relation have to be known (annotated) before it can be successfully learned and applied to the task of entity classification. Please note that with this evaluation protocol the amount of labeled entities is fixed, whereas the amount of labels on relations is varied. Secondly both tasks are measured on a separate set where no information is given.

Figure 3 shows that both task are successfully solved with about 10% of labeles on relations. This means that the proposed LRE method can easily be applied for classification tasks where only implicit relations are given.

6 Conclusion and Future Work

In this paper we have described the LRE method for learning to automatically extract relations from noisy datasets. We have shown that only a small amount

of training examples is necessary to learn a relation. We also have shown that the learned relations approximate the manually annotated relations successfully and using them for relational entity classification results in comparable performance to manually annotated relations. Thus our proposed method can replace a domain expert for the task of relation extraction from noisy data.

In future work we would like to investigate active learning techniques to select the training dataset for learning a relation.

7 Acknowledgements

This work was funded by the X-Media project (www.x-media-project.org) sponsored by the European Commission as part of the Information Society Technologies (IST) programme under EC grant number IST-FP6-026978.

References

1. Lu, Q., Getoor, L.: Link-based text classification. In: Proceedings of IJCAI Workshop on Text Mining and Link Analysis. (2003)
2. Macskassy, S.A., Provost, F.: A simple relational classifier. In: Proceedings of the Multi-relational Data Mining Workshop ACM SIGKDD. (2003)
3. Neville, J., Jensen, D., Friedland, L., Hay, M.: Learning relational probability trees. In: Proceedings of SIGKDD. (2003)
4. Fellegi, I.P., Sunter, A.B.: A theory for record linkage. Journal of the American Statistical Association **64** (1969) 1183–1210
5. Cohen, W.W., Richman, J.: Learning to match and cluster large high-dimensional data sets for data integration. In: Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD-2002), Edmonton, Alberta (2002) 475–480
6. Bilenko, M., Mooney, R.J.: Adaptive duplicate detection using learnable string similarity measures. In: Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD-2003), Washington, DC (2003)
7. Preisach, C., Rendle, S., Schmidt-Thieme, L.: Relational classification using automatically extracted relations by record linkage. In: Proceedings of the High Level Information Extraction Workshop at the European Conference on Machine Learning. (2008)
8. Preisach, C., Schmidt-Thieme, L.: Ensembles of relational classifiers. Knowledge and Information Systems (2008) 249–272
9. Cohen, W.W., Ravikumar, P., Fienberg, S.E.: A comparison of string distance metrics for name-matching tasks. In: Proceedings of the IJCAI-2003 Workshop on Information Integration on the Web, Acapulco, Mexico (August 2003) 73–78
10. Chang, C.C., Lin, C.J.: LIBSVM: a library for support vector machines. (2001) Software available at http://www.csie.ntu.edu.tw/ cjlin/libsvm.
11. Baxter, R., Christen, P., Churches, T.: A comparison of fast blocking methods for record linkage. In: Proceedings of the 2003 ACM SIGKDD Workshop on Data Cleaning, Record Linkage, and Object Consolidation, Washington, DC (2003)
12. Rendle, S., Schmidt-Thieme, L.: Scaling record linkage to non-uniform distributed class sizes. In: In Proceedings of the 12th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD 2008), Osaka (2008)