A Comparison of Propositionalization Strategies for Creating Features from Linked Open Data

Petar Ristoski and Heiko Paulheim

University of Mannheim, Germany
Research Group Data and Web Science
{petar.ristoski,heiko}@informatik.uni-mannheim.de

Abstract. Linked Open Data has been recognized as a valuable source for background information in data mining. However, most data mining tools require features in propositional form, i.e., binary, nominal or numerical features associated with an instance, while Linked Open Data sources are usually graphs by nature. In this paper, we compare different strategies for creating propositional features from Linked Open Data (a process called *propositionalization*), and present experiments on different tasks, i.e., classification, regression, and outlier detection. We show that the choice of the strategy can have a strong influence on the results.

Keywords: Linked Open Data, Data Mining, Propositionalization, Feature Generation

1 Introduction

Linked Open Data [1] has been recognized as a valuable source of background knowledge in many data mining tasks. Augmenting a dataset with features taken from Linked Open Data can, in many cases, improve the results of a data mining problem at hand, while externalizing the cost of maintaining that background knowledge [18].

Most data mining algorithms work with a propositional *feature vector* representation of the data, i.e., each instance is represented as a vector of features $\langle f_1, f_2, ..., f_n \rangle$, where the features are either binary (i.e., $f_i \in \{true, false\}$), numerical (i.e., $f_i \in \mathbb{R}$), or nominal (i.e., $f_i \in S$, where S is a finite set of symbols). Linked Open Data, however, comes in the form of *graphs*, connecting resources with types and relations, backed by a schema or ontology.

Thus, for accessing Linked Open Data with existing data mining tools, transformations have to be performed, which create propositional features from the graphs in Linked Open Data, i.e., a process called *propositionalization* [11]. Usually, binary features (e.g., `true` if a type or relation exists, `false` otherwise) or numerical features (e.g., counting the number of relations of a certain type) are used [21]. Other variants, e.g., computing the fraction of relations of a certain type, are possible, but rarely used.

Our hypothesis in this paper is that the strategy of creating propositional features from Linked Open Data may have an influence on the data mining result. For example, promiximity-based algorithms like k-NN will behave differently depending on the strategy used to create numerical features, as that strategy has a direct influence on most distance functions.

In this paper, we compare a set of different strategies for creating features from types and relations in Linked Open Data. We compare those strategies on a number of different datasets and across different tasks, i.e., classification, regression, and outlier detection.

The rest of this paper is structured as follows. Section 2 gives a brief overview on related work. In section 3, we discuss a number of strategies used for the generation of propositional features. Section 4 introduces the datasets and tasks used for evaluation, and provides a discussion of results. We conclude with a review of our findings, and an outlook on future work.

2 Related Work

In the recent past, a few approaches for propositionalizing Linked Open Data for data mining purposes have been proposed. Many of those approaches are supervised, i.e., they let the user formulate SPARQL queries, which means that they leave the propositionalization strategy up to the user, and a fully automatic feature generation is not possible. Usually, the resulting features are binary, or numerical aggregates using `SPARQL COUNT` constructs [2, 8, 9, 16, 10]. In [21], we have proposed an *unsupervised* approach allowing for both binary features and numerical aggregates.

A similar problem is handled by *Kernel functions*, which compute the distance between two data instances. They are used in kernel-based data mining and machine learning algorithms, most commonly support vector machines (SVMs), but can also be exploited for tasks such as clustering.. Several kernel functions suitable for Linked Open Data have been proposed [3, 7, 14]. While Kernel functions can be designed in a flexible manner, and support vector machines are often performing quite well on classification and regression tasks, they cannot be combined with arbitrary machine learning methods, e.g., decision tree learning.

3 Strategies

When creating features for a resource, we take into account the relation to other resources. We distinguish strategies that use the object of *specific relations*, and strategies that only take into account the presence of *relations as such*.

3.1 Strategies for Features Derived from Specific Relations

Some relations in Linked Open Data sources play a specific role. One example are `rdf:type` relations assigning a direct type to a resource. A statement `r`

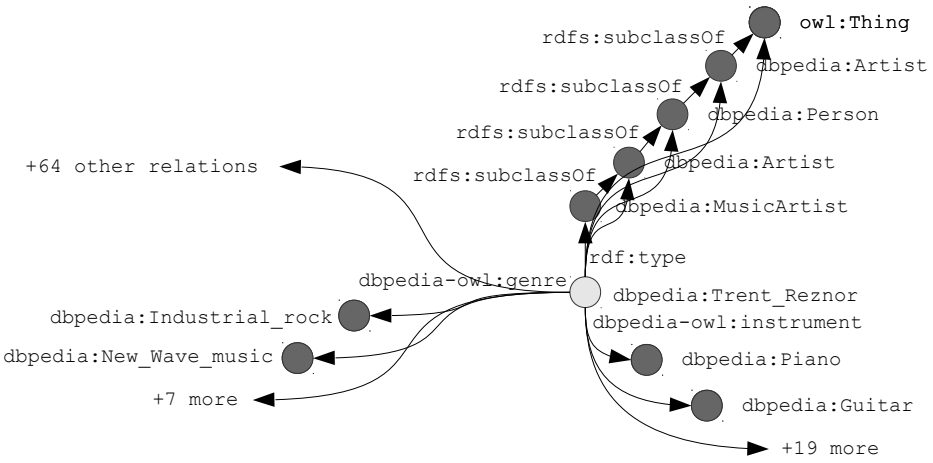

Fig. 1: Example DBpedia resource (`dbpedia:Trent_Reznor`) and an excerpt of its types and relations

`rdf:type C` is typically translated into description logics as $C(r)$, i.e., `rdf:type` is treated differently from any other predicate. For some datasets, similar relations exist, e.g., the `dcterms:subject` relations in DBpedia [13] which contain a link to the category of the original Wikipedia article a DBpedia resource is derived from.

For such relations, we propose three strategies:

- Creating a *binary feature* indicating presence or absence of the relation's object.
- Creating a *relative count feature* indicating the relative count of the relation's object. For a resource that has a relation to n objects, each feature value is $\frac{1}{n}$.
- Creating a *TF-IDF feature*, whose value is $\frac{1}{n} \cdot log \frac{N}{|\{r|C(r)\}|}$, where N is the total number of resources in the dataset, and $|\{r|C(r)\}|$ denotes the number of resources that have the respective relation r to C.

The rationale for using relative counts is that if there are only a few relations of a particular kind, each individual related object may be more important. For example, for a general book which has a hundred topics, each of those topics is less characteristic for the book than a specific book with only a few topics. Thus, that strategy takes into account both the existence and the importance of a certain relation.

The rationale for using TF-IDF is to further reduce the influence of too general features, in particular when using a distance-based mining algorithm. Table 1 shows the features generated for the example depicted in Fig.1. It can be observed that using TF-IDF implicitly gives a higher weight to more specific features, which can be important in distance-based mining algorithms (i.e., it increases the similarity of two objects more if they share a more specific type than a more abstract one).

Table 1: Features for `rdf:type` and relations as such, generated for the example shown in Fig. 1. For TF-IDF, we assume that there are 1,000 instances in the dataset, all of which are persons, 500 of which are artists, and 100 of which are music artists with genres and instruments.

	Specific relation: `rdf:type`					Relations as such	
Strategy	`MusicArtist`	`Artist`	`Person`	`Agent`	`Thing`	`genre`	`instrument`
Binary	`true`	`true`	`true`	`true`	`true`	`true`	`true`
Count	–	–	–	–	–	9	21
Relative Count	0.2	0.2	0.2	0.2	0.2	0.091	0.212
TF-IDF	0.461	0.139	0	0	0	0.209	0.488

3.2 Strategies for Features Derived from Relations as Such

Generic relations describe how resources are related to other resources. For example, a writer is connected to her birthplace, her alma mater, and the books she has written. Such relations between a resource r and a resource r' are expressed in description logics as $p(r, r')$ (for an outgoing relation) or $p(r', r)$ (for an incoming relation), where p can be any relation.

In general, we treat incoming (rel in) and outgoing (rel out) relations. For such generic relations, we propose four strategies:

- Creating a *binary feature* for each relation.
- Creating a *count feature* for each relation, specifying the number of resources connected by this relation.
- Creating a *relative count feature* for each relation, specifying the fraction of resources connected by this relation. For a resource that has total number of P outgoing relations, the relative count value for a relation $p(r, r')$ is defined as $\frac{n_p}{P}$, where n_p is the number of outgoing relations of type p. The feature is defined accordingly for incoming relations
- Creating a *TF-IDF feature* for each relation, whose value is $\frac{n_p}{P} \cdot log \frac{N}{|\{r|\exists r' : p(r,r')\}|}$, where N is the overall number of resources, and $|\{r|\exists r' : p(r, r')\}|$ denotes the number of resources for which the relation $p(r, r')$ exists. The feature is defined accordingly for incoming relations.

The rationale of using relative counts is that resources may have multiple types of connections to other entities, but not all of them are equally important. For example, a person who is mainly a musician may also have written one book, but recorded many records, so that the relations get different weights. In that case, he will be more similar to other musicians than to other authors – which is not the case if binary features are used.

The rationale of using TF-IDF again is to reduce the influence of too general relations. For example, two persons will be more similar if both of them have recorded records, rather than if both have a last name. The IDF factor accounts for that weighting. Table 1 shows the features generated from the example in Fig. 1.

4 Evaluation

We evaluated the strategies outlined above on six different datasets, two for each task of classification, regression, and outlier detection.

4.1 Tasks and Datasets

The following datasets were used in the evaluation:

- The *Auto MPG* data set[1], a dataset that captures different characteristics of cars (such as cyclinders, transmission horsepower), and the target is to predict the fuel consumption in Miles per Gallon (MPG) as a regression task [23]. Each car in the dataset was linked to the corresponding resource in DBpedia.
- The *Cities* dataset contains a list of cities and their quality of living (as a numerical score), as captured by Mercer [17]. The cities are mapped to DBpedia. We use the dataset both for regression as well as for classification, discretizing the target variable into high, medium, and low.
- The *Sports Tweets* dataset consists of a number of tweets, with the target class being whether the tweet is related to sports or not.[2] The dataset was mapped to DBpedia using DBpedia Spotlight [15].
- The *DBpedia-Peel* dataset is a dataset where each instance is a link between the DBpedia and the Peel Sessions LOD datasets. Outlier detection is used to identify links whose characteristics deviate from the majority of links, which are then regarded to be wrong. A partial gold standard of 100 links exists, which were manually annotated as right or wrong [19].
- The *DBpedia-DBTropes* dataset is a similar dataset with links between DBpedia and DBTropes.

For the classification and regression tasks, we use direct types (i.e., `rdf:type`) and DBpedia categories (i.e., `dcterms:subject`), as well as all strategies for generic relations. For the outlier detection tasks, we only use direct types and generic relations, since categories do not exist in the other LOD sources involved. An overview of the datasets, as well as the size of each feature set, is given in Table 2.

For classification tasks, we use Naïve Bayes, k-Nearest Neighbors (with k=3), and C4.5 decision tree. For regression, we use Linear Regression, M5Rules, and k-Nearest Neighbors (with k=3). For outlier detection, we use Global Anomaly Score (GAS, with k=25), Local Outlier Factor (LOF), and Local Outlier Probabilities (LoOP, with k=25). We measure accuracy for classification tasks, root-mean-square error (RMSE) for regression tasks, and area under the ROC curve (AUC) for outlier detection tasks.

[1] `http://archive.ics.uci.edu/ml/datasets/Auto+MPG`

[2] `https://github.com/vinaykola/twitter-topic-classifier/blob/master/training.txt`

Table 2: Datasets used in the evaluation. Tasks: C=Classification, R=Regression, O=Outlier Detection

Dataset	Task	# instances	# types	# categories	# rel in	# rel out	# rel in & out
Auto MPG	R	391	264	308	227	370	597
Cities	C/R	212	721	999	1,304	1,081	2,385
Sports Tweets	C	5,054	7,814	14,025	3,574	5,334	8,908
DBpedia-Peel	O	2,083	39	-	586	322	908
DBpedia-DBTropes	O	4,228	128	-	912	2,155	3,067

The evaluations are performed in RapidMiner, using the Linked Open Data extension [22]. For classification, regression, and outlier detection, we use the implementation in RapidMiner where available, otherwise, the corresponding implementations from the Weka[3] and Anomaly Detection [6] extension in RapidMiner were used. The RapidMiner processes and datasets used for the evaluation can be found online.[4] The strategies for creating propositional features from Linked Open Data are implemented in the RapidMiner Linked Open Data extension[5] [22].

4.2 Results

For each of the three tasks we report the results for each of the feature sets, generated using different propositionalization strategies. The classification and regression results are calculated using stratified 10-fold cross validation, while for the outlier detection the evaluations were made on the partial gold standard of 100 links for each of the datasets.[6]

Table 3 shows the classification accuracy for the Cities and Sports Tweets datasets. We can observe that the results are not consistent, but the best results for each classifier and for each feature set are achieved using different representation strategy. Only for the incoming relations feature set, the best results for the Cities dataset for each classifier are achieved when using the *Binary* strategy, while for the Sports Tweets dataset the best results are achieved when using *Count* strategy. We can observe that for most of the generic relation feature sets using *TF-IDF* strategy leads to poor results. That can be explained with the fact that *TF-IDF* tends to give higher weights to relations that appear rarely in the dataset, which also might be a result of erroneous data. Also, on the Cities dataset it can be noticed that when using k-NN on the incoming relations feature set, the difference in the results using different strategies is rather high.

[3] `https://marketplace.rapid-i.com/UpdateServer/faces/product_details.xhtml?productId=rmx_weka`

[4] `http://data.dws.informatik.uni-mannheim.de/propositionalization_strategies/`

[5] `http://dws.informatik.uni-mannheim.de/en/research/rapidminer-lod-extension`

[6] Note that we measure the capability of finding errors by outlier detection, not of outlier detection as such, i.e., natural outliers may be counted as false positives.

Table 3: Classification accuracy results for the Cities and Sports Tweets datasets, using Naïve Bayes(NB), k-Nearest Neighbors (k-NN, with k=3), and C4.5 decision tree (C4.5) as classification algorithms, on five different feature sets, generated using three propositionalization strategies, for *types* and *categories* feature sets, and four propositionalization strategies for the *incoming* and *outgoing relations* feature sets. The best result for each feature set, for each classification algorithm is marked in bold.

Datasets		Cities				Sports Tweets			
Features	Representation	NB	k-NN	C4.5	Avg.	NB	k-NN	C4.5	Avg.
types	Binary	.557	**.561**	**.590**	**.569**	.8100	**.829**	**.829**	.822
	Relative Count	**.571**	.496	.552	.539	.809	.814	.818	.814
	TF-IDF	**.571**	.487	.547	.535	**.821**	.824	.826	**.824**
categories	Binary	.557	**.499**	.561	.539	.822	.765	.719	.769
	Relative Count	**.595**	.443	**.589**	**.542**	**.907**	**.840**	.808	**.852**
	TF-IDF	.557	**.499**	.570	.542	.896	.819	**.816**	.844
rel in	Binary	**.604**	**.584**	**.603**	**.597**	.831	.836	.846	.838
	Count	.566	.311	.593	.490	**.832**	**.851**	**.854**	**.845**
	Relative Count	.491	.382	.585	.486	.695	.846	.851	.7977
	TF-IDF	.349	.382	.542	.424	.726	.846	.849	.8077
rel out	Binary	.476	**.600**	.567	**.547**	**.806**	.823	.844	.824
	Count	**.499**	.552	**.585**	.546	.799	.833	**.850**	**.827**
	Relative Count	.480	.584	.566	.543	.621	.842	.835	.766
	TF-IDF	.401	.547	.585	.511	.699	**.844**	.841	.7949
rel in & out	Binary	**.594**	**.585**	.564	**.581**	**.861**	.851	.864	.859
	Count	.561	.542	**.608**	.570	.860	**.860**	**.871**	**.864**
	Relative Count	.576	.471	.565	.537	.700	.845	.872	.8058
	TF-IDF	.401	.462	.584	.482	.751	.848	.861	.820

Table 4 shows the results of the regression task for the Auto MPG and Cities datasets. For the Auto MPG dataset, for M5Rules and k-NN classifiers the best results are achieved when using *Relative Count* and *TF-IDF* for all feature sets, while the results for LR are mixed. For the Cities dataset we can observe that the results are mixed for the types and categories feature set, but for the generic relations feature sets, the best results are achieved when using *Binary* representation. Also, it can be noticed that when using linear regression, there is a drastic difference in the results between the strategies.

Table 5 shows the results of the outlier detection task for the DBpedia-Peel and DBpedia-DBTropes datasets. In this task we can observe much higher difference in performances when using different propositionalization strategies. We can observe that the best results are achieved when using relative count features. The explanation is that in this task, we look at the implicit types of entities linked when searching for errors (e.g., a book linked to a movie of the same name), and those types are best characterized by the distribution of relations, as also reported in [20]. On the other hand, TF-IDF again has the tendency to assign high weights to rare features, which may also be an effect of noise.

By analyzing the results on each task, we can conclude that the chosen propositionalization strategy has major impact on the overall results. Also, in some

Table 4: Root-mean-square error (RMSE) results for the Auto MPG and Cities datasets, using Linear Regression (LR), M5Rules (M5), and k-Nearest Neighbors(k-NN, with k=3) as regression algorithms, on five different feature sets, generated using three propositionalization strategies, for *types* and *categories* feature sets, and four propositionalization strategies for the *incoming* and *outgoing relations* feature sets. The best result for each feature set, for each regression algorithm is marked in bold.

Datasets		Auto MPG				Cities			
Features	Representation	LR	M5	k-NN	Avg.	LR	M5	k-NN	Avg.
types	Binary	3.95	3.05	3.63	3.54	24.30	18.79	**22.16**	21.75
	Relative Count	**3.84**	**2.95**	**3.57**	**3.45**	18.04	19.69	33.56	23.77
	TF-IDF	3.86	2.96	**3.57**	3.46	**17.85**	**18.77**	22.39	**19.67**
categories	Binary	**3.69**	2.90	3.61	**3.40**	**18.88**	22.32	**22.67**	**21.29**
	Relative Count	3.74	2.97	3.57	3.43	18.95	**19.98**	34.48	24.47
	TF-IDF	3.78	**2.90**	**3.56**	3.41	19.02	22.32	23.18	21.51
rel in	Binary	**3.84**	2.86	3.61	**3.44**	**49.86**	**19.20**	**18.53**	**29.20**
	Count	3.89	2.96	4.61	3.82	138.04	19.91	19.2	59.075
	Relative Count	3.97	2.91	**3.57**	3.48	122.36	22.33	18.87	54.52
	TF-IDF	4.10	**2.84**	**3.57**	3.50	122.92	21.94	18.56	54.47
rel out	Binary	**3.79**	3.08	3.59	**3.49**	**20.00**	**19.36**	**20.91**	**20.09**
	Count	4.07	2.98	4.14	3.73	36.31	19.45	23.99	26.59
	Relative Count	4.09	**2.94**	**3.57**	3.53	43.20	21.96	21.47	28.88
	TF-IDF	4.13	3.00	3.57	3.57	28.84	20.85	22.21	23.97
rel in & out	Binary	3.99	3.05	3.67	3.57	**40.80**	**18.80**	**18.21**	**25.93**
	Count	3.99	3.07	4.54	3.87	107.25	19.52	18.90	48.56
	Relative Count	**3.92**	**2.98**	**3.57**	**3.49**	103.10	22.09	19.60	48.26
	TF-IDF	3.98	3.01	**3.57**	3.52	115.37	20.62	19.70	51.89

cases there is a drastic performance differences between the strategies that are used. Therefore, in order to achieve the best performances, it is important to choose the most suitable propositionalization strategy, which mainly depends on the given dataset, the given data mining task, and the data mining algorithm to be used.

When looking at aggregated results, we can see that for the classification and regression tasks, binary and count features work best in most cases. Furthermore, we can observe that algorithms that rely on the concept of *distance*, such as k-NN, linear regression, and most outlier detection methods, show a stronger variation of the results across the different strategies than algorithms that do not use distances (such as decision trees).

5 Conclusion and Outlook

Until now, the problem of finding the most suitable propositionalization strategy for creating features from Linked Open Data has not been tackled, as previous researches focused only on binary, or in some cases numerical representation of features. In this paper, we have compared different strategies for creating propositional features from types and relations in Linked Open Data. We have implemented three propositionalization strategies for specific relations, like `rdf:type`

Table 5: Area under the ROC curve (AUC) results for the DBpedia-Peel and Dbpedia-DBTropes datasets, using Global Anomaly Score (GAS, with k=25), Local Outlier Factor (LOF), and Local Outlier Probabilities (LoOP, with k=25) as outlier detection algorithms, on four different feature sets, generated using three propositionalization strategies, for *types* feature set, and four propositionalization strategies for the *incoming* and *outgoing relations* feature sets. The best result for each feature set, for each outlier detection algorithm is marked in bold.

Datasets		DBpedia-Peel				DBpedia-DBTropes			
Features	Representation	GAS	LOF	LoOP	Avg.	GAS	LOF	LoOP	Avg.
types	Binary	**0.386**	0.486	0.554	0.476	**0.503**	0.627	**0.605**	**0.578**
	Relative Count	0.385	0.398	0.595	0.459	**0.503**	0.385	0.314	0.401
	TF-IDF	**0.386**	**0.504**	**0.602**	**0.497**	**0.503**	**0.672**	0.417	0.531
rel in	Binary	0.169	0.367	0.288	0.275	0.425	0.520	0.450	0.465
	Count	0.200	0.285	0.290	0.258	0.503	**0.590**	**0.602**	**0.565**
	Relative Count	**0.293**	**0.496**	**0.452**	**0.414**	**0.589**	0.555	0.493	0.546
	TF-IDF	0.140	0.353	0.317	0.270	0.509	0.519	0.568	0.532
rel out	Binary	0.250	0.195	0.207	0.217	0.325	0.438	0.432	0.398
	Count	0.539	0.455	**0.391**	0.462	0.547	0.577	**0.522**	0.549
	Relative Count	**0.542**	**0.544**	**0.391**	**0.492**	**0.618**	0.601	0.513	**0.577**
	TF-IDF	0.116	0.396	0.240	0.251	0.322	**0.629**	0.471	0.474
rel in & out	Binary	0.324	0.430	0.510	0.422	0.351	0.439	0.396	0.396
	Count	0.527	0.367	0.454	0.450	0.565	0.563	0.527	0.553
	Relative Count	**0.603**	**0.744**	**0.616**	**0.654**	**0.667**	**0.672**	**0.657**	**0.665**
	TF-IDF	0.202	0.667	0.483	0.451	0.481	0.462	0.500	0.481

and `dcterms:subject`, and four strategies for generic relations. We conducted experiments on six different datasets, across three different data mining tasks, i.e. classification, regression and outlier detection. The experiments show that the chosen propositionalization strategy might have a major impact on the overall results. However, it is difficult to come up with a general recommendation for a strategy, as it depends on the given data mining task, the given dataset, and the data mining algorithm to be used.

For future work, additional experiments can be performed on more feature sets. For example, a feature sets of qualified incoming and outgoing relation can be generated, where qualified relations attributes beside the type of the relation take the type of the related resource into account. The evaluation can be extended on more datasets, using and combining attributes from multiple Linked Open Data sources. Also, it may be interesting to examine the impact of the propositionalization strategies on even more data mining tasks, such as clustering and recommender systems.

So far, we have considered only statistical measures for feature representation without exploiting the semantics of the data. More sophisticated strategies that combine statistical measures with the semantics of the data can be developed. For example, we can represent the connection between different resources in the graph by using some of the standard properties of the graph, such as the depth of the hierarchy level of the resources, the fan-in and fan-out values of the resources, etc.

The problem of propositionalization and feature weighting has been extensively studied in the area of text categorization [4, 12]. Many approaches have been proposed, which can be adapted and applied on Linked Open Data datasets. For example, adapting supervised weighting approaches, such as [5, 25], might resolve the problem with the erroneous data when using TF-IDF strategy.

Furthermore, some of the statistical measures can be used as feature selection metrics when extracting data mining features from Linked Open Data. For example, considering the semantics of the resources, the IDF value can be computed upfront for all feature candidates, and can be used for selecting the most valuable features before the costly feature generation. Thus, intertwining propositionalization and feature selection strategies for Linked Open Data [24] will be an interesting line of future work.

In summary, this paper has revealed some insights in a problem largely overlooked so far, i.e., choosing different propositionalization for mining Linked Open Data. We hope that these insights help researchers and practicioners in designing methods and systems for mining Linked Open Data.

Acknowledgements

The work presented in this paper has been partly funded by the German Research Foundation (DFG) under grant number PA 2373/1-1 (Mine@LOD).

References

1. Christian Bizer, Tom Heath, and Tim Berners-Lee. Linked Data - The Story So Far. *International Journal on Semantic Web and Information Systems*, 5(3):1–22, 2009.
2. Weiwei Cheng, Gjergji Kasneci, Thore Graepel, David Stern, and Ralf Herbrich. Automated feature generation from structured knowledge. In *20th ACM Conference on Information and Knowledge Management (CIKM 2011)*, 2011.
3. Gerben Klaas Dirk de Vries and Steven de Rooij. A fast and simple graph kernel for rdf. In *Proceedings of the Second International Workshop on Knowledge Discovery and Data Mining Meets Linked Open Data*, 2013.
4. Zhi-Hong Deng, Shi-Wei Tang, Dong-Qing Yang, Ming Zhang Li-Yu Li, and Kun-Qing Xie. A comparative study on feature weight in text categorization. In *Advanced Web Technologies and Applications*, pages 588–597. Springer, 2004.
5. Jyoti Gautam and Ela Kumar. An integrated and improved approach to terms weighting in text classification. *IJCSI International Journal of Computer Science Issues*, 10(1), 2013.
6. Markus Goldstein. Anomaly detection. In *RapidMiner – Data Mining Use Cases and Business Analytics Applications*. 2014.
7. Yi Huang, Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. A scalable kernel approach to learning in semantic graphs with applications to linked data. In *1st Workshop on Mining the Future Internet*, 2010.
8. Venkata Narasimha Pavan Kappara, Ryutaro Ichise, and O.P. Vyas. Liddm: A data mining system for linked data. In *Workshop on Linked Data on the Web (LDOW2011)*, 2011.

9. Mansoor Ahmed Khan, Gunnar Aastrand Grimnes, and Andreas Dengel. Two pre-processing operators for improved learning from semanticweb data. In *First RapidMiner Community Meeting And Conference (RCOMM 2010)*, 2010.
10. Christoph Kiefer, Abraham Bernstein, and André Locher. Adding data mining support to sparql via statistical relational learning methods. In *Proceedings of the 5th European Semantic Web Conference on The Semantic Web: Research and Applications*, ESWC'08, pages 478–492, Berlin, Heidelberg, 2008. Springer-Verlag.
11. Stefan Kramer, Nada Lavrač, and Peter Flach. Propositionalization approaches to relational data mining. In Sašo Džeroski and Nada Lavrač, editors, *Relational Data Mining*, pages 262–291. Springer Berlin Heidelberg, 2001.
12. Man Lan, Chew-Lim Tan, Hwee-Boon Low, and Sam-Yuan Sung. A comprehensive comparative study on term weighting schemes for text categorization with support vector machines. In *Special Interest Tracks and Posters of the 14th International Conference on World Wide Web*, WWW '05, pages 1032–1033, New York, NY, USA, 2005. ACM.
13. Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N. Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick van Kleef, Sören Auer, and Christian Bizer. DBpedia – A Large-scale, Multilingual Knowledge Base Extracted from Wikipedia. *Semantic Web Journal*, 2013.
14. Uta Lösch, Stephan Bloehdorn, and Achim Rettinger. Graph kernels for rdf data. In *The Semantic Web: Research and Applications*, pages 134–148. Springer, 2012.
15. Pablo N. Mendes, Max Jakob, Andrés García-Silva, and Christian Bizer. Dbpedia spotlight: Shedding light on the web of documents. In *Proceedings of the 7th International Conference on Semantic Systems (I-Semantics)*, 2011.
16. Jindřich Mynarz and Vojtěch Svátek. Towards a benchmark for lod-enhanced knowledge discovery from structured data. In *Proceedings of the Second International Workshop on Knowledge Discovery and Data Mining Meets Linked Open Data*, pages 41–48. CEUR-WS, 2013.
17. Heiko Paulheim. Generating possible interpretations for statistics from linked open data. In *9th Extended Semantic Web Conference (ESWC)*, 2012.
18. Heiko Paulheim. Exploiting linked open data as background knowledge in data mining. In *Workshop on Data Mining on Linked Open Data*, 2013.
19. Heiko Paulheim. Identifying wrong links between datasets by multi-dimensional outlier detection. In *Workshop on Debugging Ontologies and Ontology Mappings (WoDOOM), 2014*, 2014.
20. Heiko Paulheim and Christian Bizer. Type inference on noisy rdf data. In *International Semantic Web Conference*, pages 510–525, 2013.
21. Heiko Paulheim and Johannes Fürnkranz. Unsupervised Generation of Data Mining Features from Linked Open Data. In *International Conference on Web Intelligence, Mining, and Semantics (WIMS'12)*, 2012.
22. Heiko Paulheim, Petar Ristoski, Evgeny Mitichkin, and Christian Bizer. Data mining with background knowledge from the web. In *RapidMiner World*, 2014. To appear.
23. J. Ross Quinlan. Combining instance-based and model-based learning. In *ICML*, page 236, 1993.
24. Petar Ristoski and Heiko Paulheim. Feature selection in hierarchical feature spaces. In *Discovery Science*, 2014. To appear.
25. Pascal Soucy and Guy W. Mineau. Beyond tfidf weighting for text categorization in the vector space model. In *Proceedings of the 19th International Joint Conference on Artificial Intelligence*, IJCAI'05, pages 1130–1135, San Francisco, CA, USA, 2005. Morgan Kaufmann Publishers Inc.