Ilaria Tiddi Mathieu d'Aquin Nicolas Jay (Eds.)

LD4KD2014
Linked Data for Knowledge Discovery

Co-located with European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases

Nancy, France, Spetember 19th, 2014

Proceedings

Editors' addresses:
Knowledge Media Institute
The Open University
Walton Hall
Milton Keynes MK76AA
United Kingdom

{ilaria.tiddi | mathieu.daquin}@open.ac.uk

Orapilleur Team
LORIA
Campus scientifique
BP 239
54506 Vandoeuvre-lès-Nancy Cedex
France
nicolas.jay@loria.fr

Preface

Linked Data have attracted a lot of attention from both developers and researchers in recent years, as the underlying technologies and principles provide new ways, following the Semantic Web standards, to overcome typical data management and consumption issues such as reliability, heterogeneity, provenance or completeness. Many areas of research have adopted these principles both for the management and dissemination of their own data and for the combined reuse of external data sources. However, the way in which Linked Data can be applicable and beneficial to the Knowledge Discovery in Databases (KDD) process is still not completely understood.

The Linked Data 4 Knowledge Discovery workshop (LD4KD), co-located within the ECML/PKDD2014 conference in Nancy (France), explores the benefits of Linked Data for the very well established KDD field. Beyond addressing the traditional data management and consumption KDD issues from a Semantic Web perspective, the workshop aims at revealing new challenges that can emerge from joining the two fields.

In order to create opportunities for communication as well as collaboration channels, the workshop accepted 8 research papers from practitioners of both fields. The first observation one can make from those contributions is that the most obvious scenario for using Linked Data in a Knowledge Discovery process is the representation of the underlying data following Semantic Web standards, as shown in [De Clercq et al.], [Bloem et al.] and [Krompass et al.], with the aim of simplifying the knowledge extraction process. With that said, other contributions targeted other aspects of KDD, such as data pre-processing or pattern interpretation, with the purpose of showing that KDD processes can benefit from including elements of Linked Data. For the purpose of data preparation, [Rabatel et al.] focuses on mining Linked Data sources, while [Zogała-Siudem et al., Ristoski et al.] use Linked Data to enrich and integrate local data. The interpretation step of the KDD process is also addressed, in the work of [Alam et al.] on results interpretation and the one of [Peña et al.] on visualisation.

We thank the authors for their submissions and the program committee for their hard work. We sincerely hope that this joint work will provide new ideas for interactions between those two, mostly isolated communities.

September 2014 Ilaria Tiddi, Mathieu d'Aquin and Nicolas Jay

Organizing Committee

Claudia D'Amato, University of Bari
Nicola Fanizzi, University of Bari
Johannes Fürnkranz, TU Darmstadt
Nathalie Hernandez, IRIT
Agnieszka Lawrynowicz, University of Poznan
Francesca Lisi, University of Bari
Vanessa Lopez, IBM Dublin
Amedeo Napoli, LORIA
Andriy Nikolov, Fluid Operations, Germany
Heiko Paulheim, University of Mannheim
Sebastian Rudolph, TU Dresden
Harald Sack, University of Potsdam
Vojtěch Svátek, University Prague
Isabelle Tellier, University of Paris
Cassia Trojahn, IRIT
Tommaso di Noia, Politecnico of Bari
Jürgen Umbrich, WU Vienna
Gerhard Wohlgenannt, WU Vienna

Program Committee

Ilaria Tiddi, Knowledge Media Institute, The Open University
Mathieu d'Aquin, Knowledge Media Institute, The Open University
Nicolas Jay, Orpailleur, Loria

Contents

A Comparison of Propositionalization Strategies for Creating Features from Linked Open Data
Petar Ristoski and Heiko Paulheim **6**

Fast Stepwise Regression on Linked Data
Barbara Zogała-Siudem and Szymon Jaroszewicz **17**

Contextual Itemset Mining in DBpedia
Julien Rabatel, Madalina Croitoru, Dino Ienco and Pascal Poncelet **27**

Identifying Disputed Topics in the News
Orphée De Clercq, Sven Hertling, Veronique Hoste, Simone Paolo Ponzetto and Heiko Paulheim **37**

Lattice-Based Views over SPARQL Query Results
Mehwish Alam and Amedeo Napoli **49**

Visual Analysis of a Research Group's Performance thanks to Linked Open Data
Oscar Peña, Jon Lázaro, Aitor Almeida, Pablo Orduña, Unai Aguilera and Diego López-De-Ipiña **59**

Machine Learning on Linked Data, a Position Paper
Peter Bloem and Gerben De Vries **69**

Probabilistic Latent-Factor Database Models
Denis Krompass, Xueyan Jiang, Maximilian Nickel and Volker Tresp **74**

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`

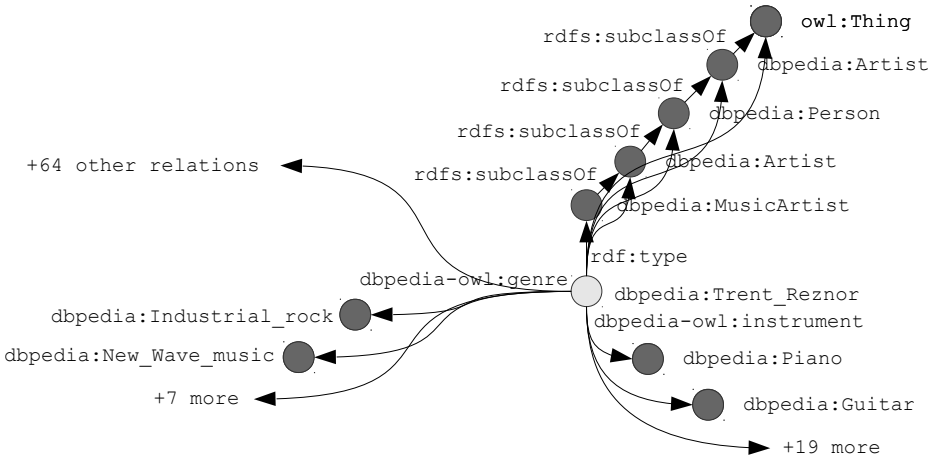


Fig. 1: Example DBpedia resource (`dbpedia:Trent_Reznor`) and an excerpt of its types and relations

`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.

Fast stepwise regression on linked data

Barbara Żogała-Siudem[1,2] and Szymon Jaroszewicz[2,3]

[1] Systems Research Institute, Polish Academy of Sciences
Warsaw, Poland
zogala@ibspan.waw.pl
[2] Institute of Computer Science, Polish Academy of Sciences
Warsaw, Poland
s.jaroszewicz@ipipan.waw.pl
[3] National Institute of Telecommunications
Warsaw, Poland

Abstract. The main focus of research in machine learning and statistics is on building more advanced and complex models. However, in practice it is often much more important to use the right variables. One may hope that recent popularity of open data would allow researchers to easily find relevant variables. However current linked data methodology is not suitable for this purpose since the number of matching datasets is often overwhelming. This paper proposes a method using correlation based indexing of linked datasets which can significantly speed up feature selection based on classical stepwise regression procedure. The technique is efficient enough to be applied at interactive speed to huge collections of publicly available linked open data.

Keywords: stepwise feature selection, linked open data, spatial indexing

1 Introduction

It is well known from statistical modeling practice that including the right variables in the model is often more important than the type of model used. Unfortunately analysts have to rely on their experience and/or intuition as there are not many tools available to help with this important task.

The rising popularity of linked open data could offer a solution to this problem. The researcher would simply link their data with other variables downloaded from a public database and use them in their model. Currently, several systems exist which allow for automatically linking publicly available data ([2, 5, 11, 17, 18]). Unfortunately, those systems are not always sufficient. Consider, for example, a researcher who wants to find out which factors affect some variable available for several countries for several consecutive years. The researcher could then link publicly available data (from, say, Eurostat [1] or the United Nations [6]) by country and year to the target modeling variable and build a linear regression model using some method of variable selection. Unfortunately, such an approach is not practical since there are literally millions of variables

available from Eurostat alone and most of them can be linked by country and year. As a result, several gigabytes of data would have to be downloaded and used for modeling.

This paper proposes an alternative approach: linking a new variable is performed not only by some key attributes but also based on the correlation with the target variable. We describe how to use spatial indexing techniques to find correlated variables quickly. Moreover, we demonstrate how such an index can be used to build stepwise regression models commonly used in statistics.

To the best of our knowledge no current system offers such functionality. The closest to the approach proposed here is the Google Correlate service [14]. It allows the user to submit a time series and find Google query whose frequency is most correlated with it. However Google Correlate is currently limited to search engine query frequencies and cannot be used with other data such as publicly available government data collections. Moreover it allows only for finding a single correlated variable, while the approach proposed here allows for automatically building full statistical models. In other words our contribution adds a statistical model construction step on top of a correlation based index such as Google Correlate.

There are several approaches to speeding up variable selection in stepwise regression models such as streamwise regression [22] or VIF regression [13]. None of them is, however, capable of solving the problem considered here: allowing an analyst to build a model automatically selecting from millions of available variables at interactive speeds.

Let us now introduce the notation. We will not make a distinction between a random variable and a vector of data corresponding to it. Variables/vectors will be denoted with lowercase letters x, y; $\bar{x}$ is the mean of x and $\mathrm{cor}(x, y)$ correlation between x and y. Matrices (sets of variables) will be denoted with boldface uppercase letters, e.g. $\mathbf{X}$. The identity matrix is denoted by $\mathbf{I}$ and $\mathbf{X}^T$ is the transpose of the matrix $\mathbf{X}$.

2 Finding most correlated variables. Multidimensional indexing

The simplest linear regression model we may think of is a model with only one variable: the one which is most correlated with the target. An example system building such models in the open data context is the Google Correlate tool [3, 14, 21]. It is based on the fact that finding a variable with the highest correlation is equivalent to finding a nearest neighbor of the response variable after appropriate normalization of the vectors.

In this paper we will normalize all input vectors (potential variables to be included in the model) as $x' = \frac{x-\bar{x}}{\|x-\bar{x}\|}$. That way, each vector is centered at zero and has unit norm, so we can think of them as of points on an $(n-1)$-sphere. It is easy to see that the correlation coefficient of two vectors x, y is simply equal

to the dot product of their normalized counterparts

$$\mathrm{cor}(x, y) = \langle x', y' \rangle.$$

Note that our normalization is slightly different from the one used in [14], but has the advantage that standard multidimensional indices can be used. After normalization the Euclidean distance between two vectors is directly related to their correlation coefficient

$$\|x - y\| = \sqrt{2 - 2\mathrm{cor}(x, y)}. \tag{1}$$

The above equation gives us a tool to quickly find variables most correlated with a given variable, which is simply the one which is closest to it in the usual geometrical sense. Moreover to find all variables x whose correlation with y is at least η one needs to find all x's for which $\|x - y\| \leqslant \sqrt{2 - 2\eta}$.

The idea now is to build an index containing all potential variables and use that index to find correlated variables quickly. Thanks to the relationship with Euclidean distance, multidimensional indexing can be used for the purpose. Building the index may be time consuming, but afterwards, finding correlated variables should be very fast. We now give a brief overview of the indexing techniques.

Multidimensional indexing. Multidimensional indices are data structures designed to allow for rapidly finding nearest neighbors in n-dimensional spaces. Typically two types of queries are supported. *Nearest neighbor queries* return k vectors in the index which are closest to the supplied query vector. Another type of query is *range query* which returns all vectors within a given distance from the query.

Due to space limitations we will not give an overview of multidimensional techniques, see e.g. [19]. Let us only note that Google Correlate [21] uses a custom designed technique called Asymmetric Hashing [8].

In the current work we use Ball Trees [12] implemented in the Python `Scikit.Learn` package. Ball Trees are supposed to work well even for high dimensional data and return exact solutions to both nearest neighbor and range queries. For faster, approximate searches we use the randomized kd-trees implemented in the FLANN package [15] (see also [16]).

Of course finding most correlated variables has already been implemented by Google Correlate. In the next section we extend the technique to building full regression models, which is the main contribution of this paper.

3 Stepwise and stagewise regression

In this section we will describe classical modeling techniques: stagewise and stepwise linear regression and show how they can be efficiently implemented in the open data context using a multidimensional index.

Stagewise regression is a simple algorithm for variable selection in a regression model which does not take into account interactions between predictor variables,

see e.g. [10] for a discussion. The algorithm is shown in Figure 1. The idea is simple: at each step we add the variable most correlated with the residual of the current model. The initial residual is the target variable y and the initial model matrix $\mathbf{X}$ contains just a column of ones responsible for the intercept term. The matrix $\mathbf{H_X} = \mathbf{X}(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T$ is the projection matrix on $\mathbf{X}$, see [9] for details.

Algorithm: Stagewise

1) $r \leftarrow y$; $\mathbf{X} \leftarrow (1, 1, \ldots, 1)^T$,
2) Find a variable x_i most correlated with r,
3) Add x_i to the model: $\mathbf{X} \leftarrow [\mathbf{X}|x_i]$,
4) Compute the new residual vector $r = y - \mathbf{H_X}y$,
5) **If** the model has improved: **goto** 2.

Fig. 1. The stagewise regression algorithm.

The stopping criterion in step 5 is based on the residual sum of squares: $RSS = r^T r = \|r\|^2$, where r is the vector of residuals (differences between true and predicted values). The disadvantage of RSS is that adding more variables can only decrease the criterion. To prevent adding too many variables to the model additional penalty terms are included, the two most popular choices are Akaike's AIC [4] and Schwarz's BIC [20]. Here we simply set a hard limit on the number of variables included in the model.

The advantage of stagewise regression is its simplicity, one only needs to compute the correlation of all candidate variables with the residual r. Thanks to this, one can easily implement stagewise regression using techniques from Section 2, so the approach can trivially be deployed in the proposed setting.

The disadvantage of stagewise regression is that is does not take into account correlations between the new variable and variables already present in the model. Consider an example dataset given in Table 1. The dataset has three predictor variables x_1, x_2, x_3 and a target variable y. The data follows an exact linear relationship: $y = 3x_1 + x_2$. It can be seen that the variable most correlated with y is x_1, which will be the first variable included in the model. The residual vector of that model, denoted r_1, is also given in the table. Clearly the variable most correlated with r_1 is x_3 giving a model $y = \beta_0 + \beta_1 x_1 + \beta_2 x_3$. But the true model does not include x_3 at all! The reason is that x_3 is highly correlated with x_1, and this correlation is not taken into account by stagewise regression.

Table 1. Example showing the difference between stepwise and stagewise regression.

x_1	x_2	x_3	y	r_1
0.03	-0.12	0.75	-0.03	0.51
-0.54	-0.10	-0.47	-1.71	-0.15
0.13	-1.03	0.11	-0.64	-0.27
0.73	-1.58	0.00	0.61	-0.09

An improvement on stagewise regression is *stepwise regression* proposed in 1960 by Efroymson [7]. The algorithm is given in Figure 2. The main idea is that at each step we add each variable to the model, compute the actual residual sum of squares (which takes into account correlations between variables) and add the variable which gives the best improvement.

Algorithm: Stepwise

1) $r \leftarrow y$; $\mathbf{X} \leftarrow (1, 1, \ldots, 1)^T$,
2) **For** each variable x_i:
   compute the residual of the model obtained by adding x_i to the current model: $r_i = y - \mathbf{H}_{[\mathbf{X}|x_i]}y$
3) Find x_{i^*}, where $i^* = \arg\min {r_i}^T r_i$,
4) **If** model: $[\mathbf{X}|x_{i^*}]$ is better than $\mathbf{X}$:
   1. Add x_{i^*} to the model $\mathbf{X} \leftarrow [\mathbf{X}|x_{i^*}]$
   2. **goto** 2).

Fig. 2. The stepwise regression algorithm.

In the example stepwise regression will choose the correct variables x_1 and then x_2, which is the best possible model. In general, stepwise regression builds better models than stagewise regression, but is more costly computationally. At each step we need to compute the RSS for several regression models, which is much more expensive than simply computing correlations.

4 Fast stepwise selection based on multidimensional indices

Stepwise regression is known to give good predictions, however when the number of attributes is large, it becomes inefficient; building a model consisting of many variables when we need to search through several millions of candidates, as is often the case with linked data, would be extremely time consuming, since at each step we would need to compute RSS of millions of multidimensional models.

In this section we present the main contribution of this paper, namely an approach to speed up the process using a multidimensional index. Our goal is to decrease the number of models whose RSS needs to be computed at each step through efficient filtering based on a multidimensional index. Assume that $k-1$ variables are already in a model and we want to add the k-th one. Let $\mathbf{X}_{k-1}$ denote the current model matrix. The gist of our approach is given in the following theorem.

Theorem 1. *Assume that the variables $x_1, \ldots, x_{k-1}$ currently in the model are orthogonal, i.e. ${\mathbf{X}_{k-1}}^T\mathbf{X}_{k-1} = \mathbf{I}$ and let $r = y - \mathbf{H}_{\mathbf{X}_{k-1}}y$ denote the residual vector of the current model. Consider two variables x_k and $x_{k'}$. Denote $c_{i,k} = \mathrm{cor}(x_i, x_k)$, $c_{i,k'} = \mathrm{cor}(x_i, x_{k'})$, $c_{r,k} = \mathrm{cor}(r, x_k)$, $c_{r,k'} = \mathrm{cor}(r, x_{k'})$. Let $\mathbf{X}_k = [\mathbf{X}_{k-1}|x_k]$ and $\mathbf{X}_{k'} = [\mathbf{X}_{k-1}|x_{k'}]$. Further, let $r_k = y - \mathbf{H}_{\mathbf{X}_k}y$ be the residual*

vector of the regression model obtained by adding variable x_k to the current model, and let $r_{k'}$ be defined analogously. Then, $\|r_{k'}\|^2 \leqslant \|r_k\|^2$ implies

$$\max\{|c_{1,k'}|, \ldots, |c_{k-1,k'}|, |c_{r,k'}|\} \geqslant \frac{|c_{r,k}|}{\sqrt{1 - c_{1,k}^2 - \ldots - c_{k-1,k}^2 + (k-1)c_{r,k}^2}}. \quad (2)$$

The theorem (the proof can be found in the Appendix) gives us a way to implement a more efficient construction of regression models through the stepwise procedure. Each step is implemented as follows. We first find a variable x_k which is most correlated with the current residual r. Then, using the right hand side of Equation 2 we find the lower bound for correlations of the potential new variable with the current residual and all variables currently in the model. Then, based on Equation 1, we can use k range queries (see Section 2) on the spatial index to find all candidate variables. Steps 2 and 3 of Algorithm 2 are then performed only for variables returned by those queries. Since step 2 is the most costly step of the stepwise procedure this can potentially result in huge speedups. The theorem assumes $x_1, \ldots, x_{k-1}$ to be orthogonal which is not always the case. However we can always orthogonalize them before applying the procedure using e.g. the QR factorization.

The final algorithm is given in Figure 3. It is worth noting that (when exact index is used like the Ball Tree) algorithm described in Figure 3 gives the same results as stepwise regression performed on full, joined data.

Algorithm: Fast stepwise based on multidimensional index

1) $r \leftarrow y$; $\mathbf{X} \leftarrow (1, 1, \ldots, 1)^T$
2) Find a variable x_1 most correlated with r # nearest neighbor query
3) Add x_1 to the model: $\mathbf{X} \leftarrow [\mathbf{X}|x_1]$
4) Compute the new residual vector $r = y - \mathbf{H}_\mathbf{X} y$
5) Find a variable x_k most correlated with r
6) $C \leftarrow \{x_k\}$ # the set of candidate variables
7) $\eta \leftarrow \frac{|c_{r,k}|}{\sqrt{1 - c_{1,k}^2 - \ldots - c_{k-1,k}^2 + (k-1)c_{r,k}^2}}$
8) **For** $i \leftarrow 1, \ldots, k-1$:
9) $\quad C \leftarrow C \cup$ all variables x such that $\|x - x_i\|^2 \leqslant 2 - 2\eta$ # range queries
10) $C \leftarrow C \cup$ all variables x such that $\|r - x_i\|^2 \leqslant 2 - 2\eta$ # range query
11) Find the best variable x_{i^*} in C using stepwise procedure, add it to the model
12) **If** the model has improved significantly: **goto** 4).

Fig. 3. The fast stepwise regression algorithm based on multidimensional index.

5 Experimental evaluation

We will now present an experimental evaluation of the proposed approach. First we give an illustrative example, then examine the efficiency.

5.1 An illustrative example

The following example is based on a part of the Eurostat database [1]. The response variable is the infant mortality rate in each country and the predictors are variables present in a part of the database concerning 'Population and social conditions', mainly 'Health'. The combined data set consists of 736 observations (data from 1990 till 2012 for each of the 32 European countries) and 164460 variables. We decided to select two variables for the model. Missing values in the time series were replaced with the most previous available value or with the next one if the previous did not exist.

Exact stepwise regression (produced with regular stepwise procedure or the Ball Tree index) resulted in the following two variables added to the model:

- ,,Causes of death by NUTS 2 regions - Crude death rate (per 100,000 inhabitants) for both men an women of age 65 or older, due to Malignant neoplasms, stated or presumed to be primary, of lymphoid, haematopoietic and related tissue"
- ,,Health personnel by NUTS 2 regions - number of physiotherapists per inhabitant".

The variables themselves are not likely to be directly related to the target, but are correlated with important factors. The first variable is most probably correlated with general life expectancy which reflects the efficiency of the medical system. The number of physiotherapists (second variable) is most probably correlated with the number of general health personnel per 100,000 inhabitants. Dealing with correlated variables is an important topic of the future research.

An analogous result was obtained using an approximate index implemented in the FLANN package. Due to the fact that the results are approximated, slightly different attributes were selected but the RSS remained comparable. Moreover, building the model using the Ball Tree index was almost 8 times faster than stepwise regression on full data, and using the FLANN index more than 400 times faster!

5.2 Performance evaluation

To assess performance we used a part of the Eurostat [1] database. The response variable was again the infant mortality rate and predictors came from the 'Population and social conditions' section, mainly: 'Health', 'Education and training' and 'Living conditions and welfare'. This resulted in a joined dataset consisting of 736 observations (data from 1990 till 2012 for 32 European countries) and over $200,000$ attributes.

The algorithms used in comparison are regular stepwise regression on full joined data ('regular step'), fast stepwise regression using two types of spatial indices and stepwise regression built using the algorithm in Figure 3 with spatial queries answered using brute force search ('step with no index').

The first two charts in Figure 4 show how the time to build a model with 3 or 5 variables changes with growing number of available attributes (i.e. the

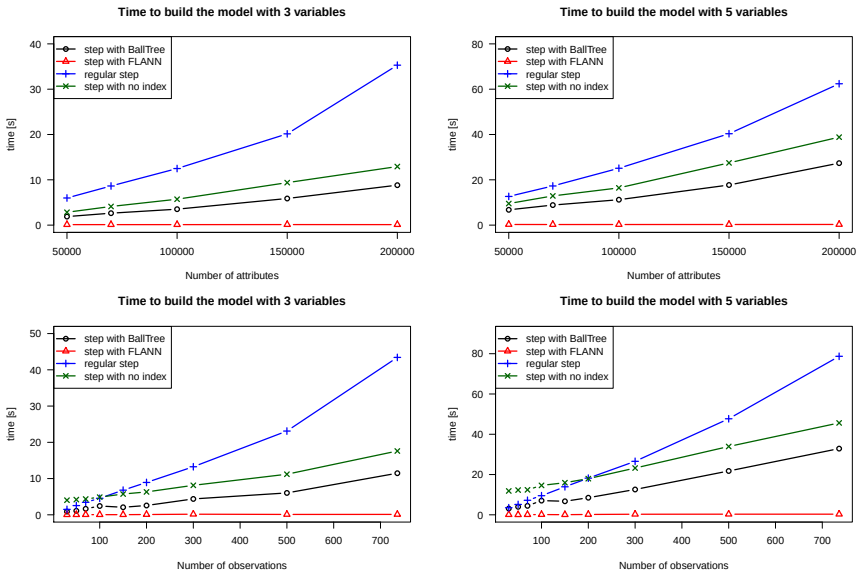


Fig. 4. Average time needed to build models with 3 or 5 variables for varying numbers of available variables and observations. 'regular step' is the classical stepwise regression, all others are the proposed fast versions using different spatial indexes and brute search.

size of full joined data). The second two charts show how the time changes with growing number of observations (records). To obtain the smaller datasets we simply drew samples of the attributes or of the observations. We can see that the best times can be obtained using FLANN index. It is worth noting that FLANN gives approximate, yet quite precise results. Slower, but still reasonably fast model construction can be obtained by using Ball Tree index, which guarantees the solution is exact. All charts show that the bigger the data the bigger the advantage from using the algorithm shown in Figure 3.

6 Conclusions

The paper presents a method for building regression model on linked open data at interactive speeds. The method is based on the use of spatial indexes for efficient finding of candidate variables. The method has been evaluated experimentally on Eurostat data and demonstrated to perform much faster than standard regression implementations.

7 Acknowledgements

The paper is co-founded by the European Union from resources of the European Social Fund. Project PO KL „Information technologies: Research and their interdisciplinary applications", Agreement UDA-POKL.04.01.01-00-051/10-00.

A Proof of Theorem 1

To prove Theorem 1 we need to introduce two lemmas.

Lemma 1. *Adding x_k to a least squares model of y based on $\mathbf{X}_{k-1} = [x_1|\ldots|x_{k-1}]$ decreases the RSS by $\frac{(x_k^T \mathbf{P}_{k-1} y)^2}{x_k^T \mathbf{P}_{k-1} x_k}$, where $\mathbf{P}_{k-1} := \mathbf{I} - \mathbf{H}_{\mathbf{X}_{k-1}}$.*

Proof. x_k can be expressed as a sum of vectors in the plane spanned by $\mathbf{X}_{k-1}$ and perpendicular to that plane: $x_k = \mathbf{H}_{\mathbf{X}_{k-1}} x_k + \mathbf{P}_{k-1} x_k$. If x_k is a linear function of columns of $\mathbf{X}_{k-1}$, adding it to the model gives no decrease of RSS, so we only need to consider $\mathbf{P}_{k-1} x_k$. It is easy to see that if x_k is uncorrelated with each column of $\mathbf{X}_{k-1}$, adding it to the model decreases RSS by $\frac{(x_k^T y)^2}{x_k^T x_k}$. This is because the RSS is then equal to $y^T \mathbf{P}_k y$, where $\mathbf{P}_k = \mathbf{P}_{k-1} - x_k (x_k^T x_k)^{-1} x_k^T$. Combining those facts with symmetry and idempotency of $\mathbf{P}_{k-1}$, RSS decreases by

$$\frac{((\mathbf{P}_{k-1} x_k)^T y)^2}{(\mathbf{P}_{k-1} x_k)^T \mathbf{P}_{k-1} x_k} = \frac{(x_k^T \mathbf{P}_{k-1}^T y)^2}{x_k^T \mathbf{P}_{k-1}^T \mathbf{P}_{k-1} x_k} = \frac{(x_k^T \mathbf{P}_{k-1} y)^2}{x_k^T \mathbf{P}_{k-1} x_k}.$$

Lemma 2. *Assume now that $\mathbf{X}_{k-1}$ is orthogonal. If adding $x_{k'}$ to the model gives lower RSS than adding x_k, then:*

$$\frac{c_{r,k}^2}{1 - c_{1,k}^2 - \ldots - c_{k-1,k}^2} < \frac{c_{r,k'}^2}{1 - c_{1,k'}^2 - \ldots - c_{k-1,k'}^2}. \tag{3}$$

Proof. From Lemma 1 we know that if $x_{k'}$ causes greater decrease in RSS then

$$\frac{(x_k^T \mathbf{P}_{k-1} y)^2}{x_k^T \mathbf{P}_{k-1} x_k} < \frac{(x_{k'}^T \mathbf{P}_{k-1} y)^2}{x_{k'}^T \mathbf{P}_{k-1} x_{k'}}.$$

We also know that (since vectors are normalized) $c_{r,k}^2 = (x_k^T r)^2 = (x_k^T \mathbf{P}_{k-1} y)^2$, and using orthogonality of $\mathbf{X}_{k-1}$ we get

$$\begin{aligned} x_k^T \mathbf{P}_{k-1} x_k &= x_k^T (\mathbf{I} - \mathbf{X}_{k-1} (\mathbf{X}_{k-1}^T \mathbf{X}_{k-1})^{-1} \mathbf{X}_{k-1}^T) x_k = x_k^T (\mathbf{I} - \mathbf{X}_{k-1} \mathbf{X}_{k-1}^T) x_k = \\ &= x_k^T x_k - (x_1^T x_k)^2 - \ldots - (x_{k-1}^T x_k)^2 = 1 - c_{1,k}^2 - \ldots - c_{k-1,k}^2, \end{aligned}$$

which proves the lemma.

Proof (of Theorem 1). If for any $i = 1, \ldots, k-1$: $|c_{i,k'}| \geqslant \frac{|c_{r,k}|}{\sqrt{1 - c_{1,k}^2 - \ldots - c_{k-1,k}^2 + (k-1) c_{r,k}^2}}$ then the inequality is true. Otherwise for all $i = 1, \ldots, k-1$:

$$|c_{i,k'}| < \frac{|c_{r,k}|}{\sqrt{1 - c_{1,k}^2 - \ldots - c_{k-1,k}^2 + (k-1) c_{r,k}^2}} \tag{4}$$

and we need to show that this implies $|c_{r,k'}| \geqslant \frac{|c_{r,k}|}{\sqrt{1 - c_{1,k}^2 - \ldots - c_{k-1,k}^2 + (k-1) c_{r,k}^2}}$. Notice first that the inequalities (4) imply

$$1 - c_{1,k'}^2 - \ldots - c_{k-1,k'}^2 > \frac{1 - c_{1,k}^2 - \ldots - c_{k-1,k}^2}{1 - c_{1,k}^2 - \ldots - c_{k-1,k}^2 + (k-1) c_{r,k}^2}.$$

Using this inequality and Lemma 2 we get the desired result:

$$c^2_{r,k'} \geqslant c^2_{r,k} \frac{1 - c^2_{1,k'} - \ldots - c^2_{k-1,k'} + c^2_{r,k'}}{1 - c^2_{1,k} - \ldots - c^2_{k-1,k} + c^2_{r,k}} > \frac{c^2_{r,k}}{1 - c^2_{1,k} - \ldots - c^2_{k-1,k} + (k-1)c^2_{r,k}}.$$

References

1. Eurostat. `http://ec.europa.eu/eurostat`.
2. Fegelod. `http://www.ke.tu-darmstadt.de/resources/fegelod`.
3. Google correlate. `http://www.google.com/trends/correlate`.
4. H. Akaike. A new look at the statistical model identification. *IEEE Transactions on Automatic Control*, AC-19(6):716–723, 1974.
5. C. Bizer, T. Heath, and T. Berners-Lee. Linked data - the story so far. *International Journal on Semantic Web and Information Systems (IJSWIS)*, 5(3):1–22, 2009.
6. United Nations Statistics Division. Undata. `http://data.un.org`.
7. M. A. Efroymson. *Multiple Regression Analysis*. Wiley, 1960.
8. A. Gersho and R. M. Gray. *Vector Quantization and Signal Compression*. Springer, 1992.
9. T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning*. Springer Series in Statistics. Springer New York Inc., New York, NY, USA, 2001.
10. T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer-Verlag, 2009.
11. T. Heath and Bizer C. *Linked Data: Evolving the Web into a Global Data Space*. Synthesis Lectures on the Semantic Web: Theory and Technology. Morgan & Claypool, 1 edition, 2011.
12. A. M. Kibriya and E. Frank. An empirical comparison of exact nearest neighbour algorithms. In *PKDD*, pages 140–151. Springer, 2007.
13. D. Lin and D. P. Foster. Vif regression: A fast regression algorithm for large data. In *ICDM '09. Ninth IEEE International Conference on Data Mining*, 2009.
14. M. Mohebbi, D. Vanderkam, J. Kodysh, R. Schonberger, H. Choi, and S. Kumar. Google correlate whitepaper. 2011.
15. M. Muja and D. Lowe. *FLANN - Fast Library for Approximate Nearest Neighbors*, 2013.
16. M. Muja and D. G. Lowe. Scalable nearest neighbor algorithms for high dimensional data. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 36, 2014.
17. H. Paulheim. Explain-a-lod: using linked open data for interpreting statistics. In *IUI*, pages 313–314, 2012.
18. H. Paulheim and J. Fürnkranz. Unsupervised generation of data mining features from linked open data. Technical Report TUD-KE-2011-2, Knowledge Engineering Group, Technische Universität Darmstadt, 2011.
19. H. Samet. *The Design and Analysis of Spatial Data Structures*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1990.
20. G. Schwarz. Estimating the dimension of a model. *The Annals of Statistics*, 6(2):461–464, 1978.
21. D. Vanderkam, R. Schonberger, H. Rowley, and S. Kumar. Technical report: Nearest neighbor search in google correlate. 2013.
22. J. Zhou, D. P. Foster, R. A. Stine, and L. H. Ungar. Streamwise feature selection. *J. Mach. Learn. Res.*, 7:1861–1885, December 2006.

Contextual Itemset Mining in DBpedia

Julien Rabatel[1], Madalina Croitoru[1], Dino Ienco[2], Pascal Poncelet[1]

[1]LIRMM, University Montpellier 2, France
[2] IRSTEA UMR TETIS, LIRMM, France

Abstract. In this paper we show the potential of contextual itemset mining in the context of Linked Open Data. Contextual itemset mining extracts frequent associations among items considering background information. In the case of Linked Open Data, the background information is represented by an Ontology defined over the data. Each resulting itemset is specific to a particular context and contexts can be related each others following the ontological structure.
We use contextual mining on DBpedia data and show how the use of contextual information can refine the itemsets obtained by the knowledge discovery process.

1 Introduction

We place ourselves in a knowledge discovery setting where we are interested in mining frequent itemsets from a RDF knowledge base. Our approach takes into account contextual data about the itemsets that can impact on what itemsets are found frequent depending on their context [16]. This paper presents a proof of concept and shows the potential advantage of *contextual* itemset mining in the Linked Open Data (LOD) setting, with respect to other approaches in the literature that *do not consider contextual information* when mining LOD data. We make the work hypothesis that the context we consider in this paper is the class type (hypothesis justified by practical interests of such consideration such as data integration, alignment, key discovery, etc.). This work hypothesis can be lifted and explored according to other contexts such as predicates, pairs of subjects and objects, etc. [1] as further discussed in Section 5.

Here we are not interested in the use of how the mined itemsets can be relevant for ontological rule discovery [15], knowledge base compression [12] etc. We acknowledge these approaches and plan to investigate how, depending on the way contexts are considered, we can mine different kind of frequent itemsets that could be further used for reasoning. Therefore, against the state of the art our contribution is:

- Taking into account contextual information when mining frequent itemsets on the Linked Open Data cloud. This allows us to refine the kind of information the mining process can provide.
- Introducing the notion of frequent contextual pattern and show how it can be exploited for algorithmic considerations.

We evaluate our approach on the DBpedia [13] dataset. In Section 2 we give a short example of the intuition of our approach. Section 3 explains the theoretical foundations of our work. In Section 4 we briefly present the DBpedia dataset and explain the obtained results. Section 5 concludes the paper.

2 Paper in a Nutshell

A semantic knowledge base is typically composed of two parts. The first part is the ontological, general knowledge about the world. Depending on the subset of first order logic used to express it this part is also called TBox (in Description Logics [4]), support (in Conceptual Graphs [8]) or rules (in Conceptual Graphs and rule based languages such as Datalog and Datalog+- [6]).

The second part is the factual knowledge about the data defining how the instances are in relation with each other. In Description Logics the factual knowledge is called ABox. Usually in the Linked Open Data, the factual knowledge is stored using RDF (usually in a RDF Triple Store) as triples "Subject Predicate Object". Recently, within the Ontology Based Data Access [7] the data (factual information) can also be stored in a relational databases.

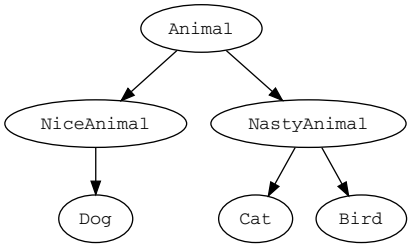
A study of the trade-off of using different storage systems (with equivalent expressivity) was recently done in [5]. DBpedia organises the data in three parts:

- The first part is the ontology (representing the TBox). The rules do not introduce existential variables in the conclusion (unlike existential rules as in Datalog+-) and they represent the Class Type hierarchy and the Predicate Type hierarchy. The ontology is guaranteed to be acyclic in the version of DBpedia we used. In the Directed Acyclic Graph (DAG) representing the ontology there are around 500 nodes, with around 400 being leaves. The maximal height is inferior to 10. The ontology we consider in the example in this section is depicted in Figure 1(b). We consider a six class ontology represented by a binary tree with height three. The algorithmic implications of the structure of the ontology are discussed in Section 5. Additionally, we consider the binary predicates "*playsWith*", "*eats*" and "*hates*" of signature "*(Animal, Animal)*".
- The second part is the mapping based types containing the instance type definition. In the example in this section, using the ontology defined in Figure 1(b), we consider the following mapping types (using a RDF triple notation): "*Bill hasType Dog*", "*Boule hasType Human*", "*Tom hasType Cat*", "*Garfield hasType Cat*" and "*Tweety hasType Bird*".
- The third part consists of the mapping based properties that correspond to the factual information. In the example here we consider the following facts: "*Bill eats Tweety*", "*Tweety hates Bill*", "*Bill playsWith Boule*", "*Tom eats Tweety*", "*Tom hates Bill*", "*Garfield hates Bill*", "*Garfield eats Tweety*", "*Garfield hates Boule*". These facts are summarized in Figure 1(a).

In this paper we make the choice of working with the data from the perspective of the Class of the Subject. As mentioned in the introduction we motivate

Subject	Predicate	Object
Bill	eats	Tweety
Tweety	hates	Bill
Bill	playsWith	Boule
Tom	eats	Tweety
Tom	hates	Bill
Garfield	hates	Bill
Garfield	eats	Tweety
Garfield	hates	Boule

(a) A fact base $\mathcal{F}$.



(b) An ontology $\mathcal{H}$.

Fig. 1: A knowledge base $KB = (\mathcal{F}, \mathcal{H})$.

tid	$\mathbf{I_{Animal}}$
$Bill$	$\{(eats, Tweety), (playsWith, Boule)\}$
$Tweety$	$\{(hates, Bill)\}$
Tom	$\{(eats, Tweety), (hates, Bill)\}$
$Garfield$	$\{(hates, Bill), (eats, Tweety), (hates, Boule)\}$

Fig. 2: The transactional database $\mathcal{T}_{KB,Animal}$ for the context *Animal* in the knowledge base KB depicted in Figure 1.

this initial choice from the perspective of various data integration tools on the Linked Open Data. One of the main challenges encountered by these tools is the mapping between various class instances. It is then not uncommon to consider the RDF database from a class type at a time.

If we consider this point of view then we model the couple "*(Predicate,Object)*" as an *item*, and the set of items associated with a given subject an *itemset*. The itemset corresponding to each distinct subject from $\mathcal{F}$ are depicted in Figure 2.

One may notice that 50% of the itemsets depicted in Figure 2 include the subset $\{(hates, Bill), (eats, Tweety)\}$, while 75% include {(hates,Bill)}.

But this is simply due to the fact that our knowledge base contains a lot of cats (that hate Bill). Actually, if we look closer, we notice that all cats hate Bill and all birds hate Bill but no dogs hate Bill. By considering this contextual information we could be more fine-tuned with respect to frequent itemsets.

3 Theoretical Foundations

The contextual frequent pattern (CFP) mining problem aims at discovering patterns whose property of being frequent is context-dependent.

This section explains the main concepts behind this notion for Linked Open Data.

We consider as input a knowledge base $KB = (\mathcal{F}, \mathcal{H})$ composed of an ontology $\mathcal{H}$ (viewed as a directed acyclic graph) and a set of facts $\mathcal{F}$.

The set of facts $\mathcal{F}$ is defined as the set of RDF triples of the form

$$(subject, predicate, object)$$

.

Each element of the triple is defined according to the ontology[1]. The ontology, also denoted the **context hierarchy** $\mathcal{H}$, is a directed acyclic graph (DAG), denoted by $\mathcal{H} = (V_\mathcal{H}, E_\mathcal{H})$, such that $V_\mathcal{H}$ is a set of vertices also called **contexts** and $E_\mathcal{H} \subseteq V_\mathcal{H} \times V_\mathcal{H}$ is a set of directed edges among contexts.

$\mathcal{H}$ is naturally associated with a partial order $<_\mathcal{H}$ on its vertices, defined as follows: given $c_1, c_2 \in V_\mathcal{H}$, $c_1 <_\mathcal{H} c_2$ if there exists a directed path from c_2 to c_1 in $\mathcal{H}$. This partial order describes a specialization relationship: c_1 is said to be more *specific* than c_2 if $c_1 <_\mathcal{H} c_2$, and more *general* than c_2 if $c_2 <_\mathcal{H} c_1$. In this case, c_2 is also called a subcontext of c_1.

A *minimal context* from $\mathcal{H}$ is a context such that no more specific context exists in $\mathcal{H}$, i.e., $c \in V_\mathcal{H}$ is minimal if and only if there is no context $c' \in V_\mathcal{H}$ such that $c' <_\mathcal{H} c$. The set of minimal contexts in $\mathcal{H}$ is denoted as $V_\mathcal{H}^-$.

Based on this knowledge base, we will build a **transactional database** such that each transaction corresponds to the set of predicates and objects of subjects of a given class. More precisely, given $KB = (\mathcal{F}, \mathcal{H})$ and $c \in V_\mathcal{H}$, the transactional database for c w.r.t. KB, denoted as $\mathcal{T}_{KB,c}$, is the set of transactions of the form $T = (tid, I_c)$ where $I_c = \{(pred, obj) | (s, pred, obj) \in \mathcal{F} \text{ and } c \text{ is the class of } s\}$.

We define $\mathcal{I}_\mathcal{H}$ as the set $\{I_c | c \in V_\mathcal{H}\}$. In this paper, we are interested in itemset mining, thus a pattern p is defined as a subset of $\mathcal{I}_\mathcal{H}$.

Definition 1 (Pattern Frequency). *Let KB be a knowledge base, p be a pattern and c be a context, the frequency of p in $\mathcal{T}_{KB,c}$ is defined as $Freq(p, \mathcal{T}_{KB,c}) = \frac{|\{(tid,I) \in \mathcal{T}_{KB,c} | p \subseteq I\}|}{|\mathcal{T}_{KB,c}|}$.*

For the sake of readability, in the rest of the paper, $Freq(p, \mathcal{T}_{KB,c})$ is denoted by $Freq(p, c)$.

Definition 2 (Contextual Frequent Pattern). *Let KB be a knowledge base, p be a pattern, c be a context and σ a mininum frequency threshold. The couple (p, c) is a **contextual frequent pattern** (CFP) in KB if:*

- *p is frequent in c, i.e., $Freq(p, c) \geq \sigma$,*
- *p is frequent in every subcontext of c, i.e., for every context c' such that $c' <_\mathcal{H} c$, $Freq(p, c') \geq \sigma$,*

[1] Given the ontology we considered in *DBPedia*, the ontology is solely composed of a class hierarchy.

Additionally, (p, c) *is* ***context-maximal*** *if there does not exist a context* C *more general than* c *such that* (p, C) *is a contextual frequent pattern.*

Definition 3. *Given a user-specified mininum frequency threshold* σ *and a knowledge base* $KB = (\mathcal{F}, \mathcal{H})$*, the contextual frequent pattern mining problem consists in enumerating all the context-maximal contextual frequent patterns in* KB*.*

Th CFP mining problem is intrinsically different from the one addressed in [17, 14]. The CFP exploits a **context hierarchy** that define relationships over the contexts associated to each transaction while in [17, 14] the taxonomic information is employed to generalize the objects over which a transaction is defined.

3.1 Algorithm for computing CFPs

The above definitions provide us with a theoretical framework for CFP mining. In the rest of this section, we design the algorithm that extracts CFPs from *DBpedia*. This algorithm is inspired from the one that was proposed in [16] for mining contextual frequent sequential patterns (i.e., a variation of the frequent itemset mining problem where itemsets are ordered within a sequence [2]). We however handle itemsets in the current study and therefore have to propose an adapted algorithm. To this end, we propose to mine CFPs through post-processing the output of a regular frequent itemset miner.

Indeed, by considering the definition of a context-maximal CFP (*cf.* Definition 2), one could imagine how to extract them via the following easy steps: *(1)* extracting frequent patterns in every context of $\mathcal{H}$ by exploiting an existing frequent itemset miner, *(2)* for each context and each frequent itemset found in this context, check whether it satisfies the requirements of a CFP (i.e., checking whether it was also found frequent in the subcontexts, and whether it is context-maximal. This approach, while convenient for its straightforwardness, is inefficient in practice. Mining every context of the hierarchy can quickly become impractical because of the number of such elements. In addition, mining all the contexts of the hierarchy is redundant, as more general contexts contain the same elements as their subcontexts.

In order to tackle these problems, we propose to remove this redundancy by mining frequent patterns in minimal contexts of $\mathcal{H}$ only and building CFPs from the patterns found frequent in those only. In consequence, we define the *decomposition* notions, by exploiting the fact that a context can be described by its minimal subcontexts in $\mathcal{H}$. To this end, we consider the *decomposition* of a context c in $\mathcal{H}$ as the set of minimal contexts in $\mathcal{H}$ being more specific than c, i.e., $decomp(c, \mathcal{H}) = \{c' \in V_{\mathcal{H}}^{-} | (c' <_{\mathcal{H}} c) \vee (c' = c)\}$. Please notice that given this definition, the decomposition of a minimal context c is the singleton $\{c\}$.

Proposition 1. *Let* KB *be a knowledge base,* p *be a pattern and* c *be a context.* (p, c) *is a contextual frequent pattern in* KB *if and only if* p *is frequent in every element of* $decomp(c)$*.*

This proposition (whose proof can be found in [16] and adapted to the current framework) is essential by allowing the reformulation of the CFP definition w.r.t. minimal contexts only: a couple (p, c) is a CFP if and only if the set of minimal contexts where p is frequent includes the decomposition of c.

Extending this property to context-maximal CFPs is straightforward. The algorithm we use to extract context-maximal CFPs in *DBpedia* data can be decomposed into the following consecutive steps:

1. **Mining.** Frequent patterns are extracted from each minimal context. At this step, by relying on Proposition 1, we do not mine non-minimal contexts. The frequent itemset miner employed to perform this step is an implementation of the *APriori* algorithm [3] provided in [10].

2. **Reading.** Output files from the previous step are read. The patterns p are indexed by the set of minimal contexts where they are frequent, denoted by l_p. Then, we initialize a hash table K as follows. The hash table keys are the sets of minimal contexts and the hash table values are the sets of patterns such that $K[l]$ contains the patterns p such that $l_p = l$. The hash table K, at the end of this step, thus stores all the patterns found frequent in at least one minimal context during the previous step. The patterns are indexed by the set of minimal contexts where they are frequent.

3. **CFP Generation.** During this step, each key l of K is passed to a routine called $maxContexts$ which performs a bottom-up traversal of the vertices of $\mathcal{H}$ in order to return the set of maximal contexts among $\{c \in V_{\mathcal{H}} \mid decomp(c) \subseteq l\}$. Such contexts satisfy the Proposition 1. Then, for each pattern p such that $l = l_p$ and each context returned by the $maxContexts$ routine, one context-maximal CFP is generated and stored. Two patterns p and p' frequent in the same minimal contexts (i.e., $l_p = l_{p'}$) are general in the same contexts. They will generate the same result via the $maxContexts$ routine. By using a hash table K to store the patterns that are frequent in the same minimal contexts, the number of calls to $maxContexts$ is greatly reduced to the number of keys in K rather than the number of distinct patterns discovered during the *mining* step.

4 Experimental Results

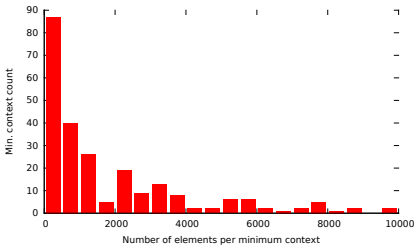
In this section, we describe the results obtained through discovering contextual frequent patterns in the *DBpedia* dataset. All experiments have been conducted on an Intel i7-3520M 2.90GHz CPU with 16 GB memory. The rest of the section is organized as follows. First, we comment the quantitative aspects of the evaluation. Second, we show and explain some examples of contextual frequent patterns found in the data.

Quantitative evaluation. In order to apply the mining algorithm to the *DBpedia* data, we pre-process them by removing all the contexts associated to less than 10

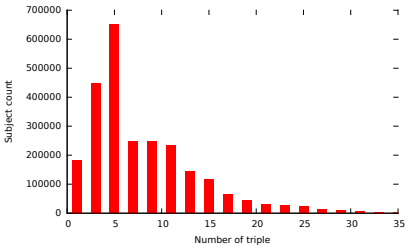
elements. The obtained contextual hierarchy contains 331 contexts, out of whic 278 are minimal. The intuition behind this pre-processing step is that extracting frequent patterns from contexts that contain a very small amount of elements is statistically insignificant and can lead to noisy results.

After the above-mentioned pre-processing, the data has the following features. The whole database contains a total of $2,501,023$ transactions. The number of elements per minimal context (i.e., the partition of the subjects over the classes) is naturally unbalanced in the *DBpedia* data. Indeed, a minimal context contains in average 8996 ± 40383 elements, with a minimum of 12 and a maximum of 577196. Figure 3(a) depicts how subjects are distributed over the minimal contexts and shows that the majority of minimal contexts contains less than 2000 transactions.

Similarly, the repartition of triples regarding their associated subjects is unbalanced. A subject in the *DBpedia* data we considered is associated in average with 7.43 ± 6.54 triples, with a maximum amount of 821 triples per subject. Please notice that this is equivalent to the average count of items per itemset in our contextual database. Figure 3(b) shows the repartition of the subjects in the data according to the number of triples they are asociated with.



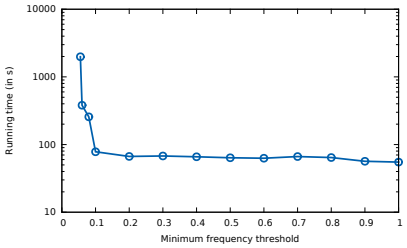
(a) Histogram depicting the repartition of elements per minimum context.



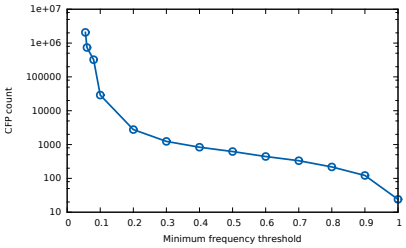
(b) Histogram depicting the repartition of triples associated to a given subject.

Fig. 3: Elements of data repartition in *DBpedia*.

Figure 4(a) shows the runtime required to discover contextual frequent patterns in the whole database according to the minimum frequency threshold. The proposed approach is shown to be scalable regarding the *DBpedia* data. The runtimes are indeed lower than 100 seconds for minimum frequency thresholds lower than 0.1. Unsurprisingly, the required time becomes much higher with low minimum frequency thresholds (around 5%). As shown in Figure 4(b), decreasing the minimum frequency threshold also provokes a higher number of discovered CFPs (more than $1,000,000$ for a minimum frequency threshold of 5%). This global behavior regarding the user-specified minimum frequency threshold is typical of frequent pattern miners.



(a) Running time w.r.t. the minimum frequency threshold.



(b) Contextual frequent pattern count w.r.t. the minimum frequency threshold.

Fig. 4: Elements of data repartition in the considered subset of *DBpedia*.

CFP examples and interpretation. CFPs have the ability to describe how (predicate,object) couples can be correlated to a class. Some examples can be found in minimal contexts of the hierarchy $\mathcal{H}$. For instance, the CFP $(\{(location, UnitedStates)\}, WineRegion)$ with a frequency of 72% in the minimal context $WineRegion$ means that "*72% of wine regions described in DBpedia are located in the United States*". Similarly, the CFP $(\{(BirthPlace, England)\}, DartsPlayer)$ with a frequency of 33% in the minimal context $DartsPlayer$ shows that "*33% of the darts players in DBpedia were born in England*". Hence, mining CFPs has the ability to describe frequent patterns in every minimal context of $\mathcal{H}$. Such CFPs, because they are associated to minimal contexts, bring other information. All extracted CFPs are context-maximal (*cf.* Definition 3). As a consequence, they also bring an additional piece of information to help an expert interpretating the results. For instance, $(\{(BirthPlace, England)\}, DartsPlayer)$ being context-maximal also indicates that $(\{(BirthPlace, England)\}, Athlete)$ is not a CFP. In other terms, the fact that the itemset $\{(BirthPlace, England)\}$ is frequent in the context $DartsPlayer$ does not hold in all the other subcontexts of $Athlete$ (such subcontexts include $TennisPlayer$, $Wrestler$, $Cyclist$, etc.).

Previous examples describe facts associated to minimal contexts only. However, the contextual frequent pattern mining problem also aims at describing how such facts can be lifted to more general contexts. A simple example can be found in the context $MusicGroup$, which has two subcontexts $Band$ and $MusicalArtist$. With a minimum frequency threshold of 10%, the context-maximal CFP $(\{hometown, UnitedStates\}, MusicGroup)$ is discovered. This pattern brings several pieces of information:

- more than 10% of $MusicGroup$ from *DBpedia*, i.e., bands or musical artists, have their hometown in the United States. More precisely, the algorithm also provides us with the exact frequency of this pattern in $MusicGroup$, i.e., 15.3%;
- this fact also holds for subcontexts of $MusicGroup$: more than 10% of bands and more than 10% of musical artists have their hometown in the United States;

– no context in $\mathcal{H}$ more general than $MusicGroup$ can be associated with the itemset $\{hometown, UnitedStates\}$ to form a CFP.

CFPs hence have the ability to describe the facts contained in *DBpedia* regarding their frequency, but also how the property of being frequent can be generalized or not in the whole context hierarchy.

5 Discussion

RDF data constitutes a rich source of information and, recently, data mining community starts to adapt its methods to extract knowledge from such kind of data [18]. In particular, preliminary works in this direction employ pattern mining techniques in order to extract frequent correlation and meta information from RDF dataset. In [9] the authors propose to use association rule mining (not using any contextual information) in order to enrich RDF schema with property axioms. The properties axioms are automatically induced by means of a pattern mining step. These axioms can be directly used as meta-data to index the RDF dataset.

[12] exploits pattern mining in order to compress RDF data. In this work the authors apply well known association rule mining algorithms to induce rules that cover the information in the RDF dataset. Once the rules are extracted, the RDF dataset is compressed considering the set of rules plus the not covered RDF triples. Both previous approaches did not employ any additional background knowledge (such as taxonomy). This information can be useful in order to exploit existing relationships among the data.

A first attempt in this direction is presented in [11]. In this work an RDF triple is an item and taxonomical information is directly employed to generalise subjects or objects at item level. Differently from this strategy, our approach allows to characterise (by means of the extracted itemsets) a node of the taxonomy supplying cues about how the extracted knowledge can be organised for its analysis. Since the taxonomy nodes we exploit (the contexts) are classes we could have presented our contribution directly from an RDF class view point. This choice would have meant that the conceptual link with the notion of context in the data mining setting was lost. Thus we preferred to keep the context terminology. Let us mention here that the shape of the ontology changes the efficiency of the algorithm. Due to the nature of our algorithm and pruning method we have better results if the DAG is not large but has a big height. We plan to apply our method to more specialised ontologies where such property is satisfied.

As explained above the mined itemsets could be used for inference rules. For instance, if in the context Cat the itemset $\{hates, Bill\}$ is frequent we can view this as a rule $\forall x Cat(x) \rightarrow hates(x, Bill)$ that hold in the knowledge base at a given time with a given confidence. We are currently working towards providing such itemsets with logical semantics and see what the frequency means in this case.

Another issue when considering the itemsets as rules is how to deal with their number, and how to order them in order to be validated by an expert.

One possible avenue we can explore is to generalise instances to concepts and try to extract less rules but more meaningful. For example we can generalise Bill to Dog and mine the rule $\forall x Cat(x) \rightarrow \exists y Dog(y) \wedge hates(x, y)$. The question is where to put the tradeoff between expressivity and number.

Finally let us mention that changing the context (not considering the class as a context but the couple (subject, object)) we could mine interesting rules about predicates. For instance we could get that $\forall x \forall y Cat(x) \wedge Dog(y) \rightarrow hates(x, y)$.

References

1. Z. Abedjan and F. Naumann. Context and target configurations for mining rdf data. In *Workshop on Search and mining entity-relationship data*, pages 23–24. ACM, 2011.
2. R. Agrawal and R. Srikant. Mining sequential patterns. In *ICDE*, pages 3–14. IEEE, 1995.
3. R. Agrawal, R. Srikant, et al. Fast algorithms for mining association rules. In *VLDB*, volume 1215, pages 487–499, 1994.
4. F. Baader. *The description logic handbook: theory, implementation, and applications*. Cambridge university press, 2003.
5. J. Baget, M. Croitoru, and B. P. L. Da Silva. Alaska for ontology based data access. In *ESWC (Satellite Events)*, pages 157–161. Springer, 2013.
6. A. Cali, G. Gottlob, T. Lukasiewicz, B. Marnette, and A. Pieris. Datalog+/-: A family of logical knowledge representation and query languages for new applications. In *LICS*, pages 228–242. IEEE, 2010.
7. D. Calvanese, G. De Giacomo, D. Lembo, M. Lenzerini, A. Poggi, and R. Rosati. Ontology-based database access. In *SEBD*, pages 324–331, 2007.
8. M. Chein and M. Mugnier. *Graph-based knowledge representation: computational foundations of conceptual graphs*. Springer, 2008.
9. D. Fleischhacker, J. Völker, and H. Stuckenschmidt. Mining rdf data for property axioms. In *OTM Conferences (2)*, pages 718–735, 2012.
10. P. Fournier-Viger, A. Gomariz, A. Soltani, H. Lam, and T. Gueniche. Spmf: Open-source data mining platform. http://www.philippe-fournier-viger.com/spmf, 2014.
11. T. Jiang and A. Tan. Mining rdf metadata for generalized association rules: knowledge discovery in the semantic web era. In *WWW*, pages 951–952, 2006.
12. A. K. Joshi, P. Hitzler, and G. Dong. Logical linked data compression. In *ESWC*, pages 170–184, 2013.
13. J. Lehmann, R. Isele, M. Jakob, A. Jentzsch, D. Kontokostas, P. N. Mendes, S. Hellmann, M. Morsey, P. van Kleef, S. Auer, and C. Bizer. DBpedia - a large-scale, multilingual knowledge base extracted from wikipedia. *Semantic Web Journal*, 2014.
14. R. A. Lisi and D. Malerba. Inducing multi-level association rules from multiple relations. *Machine Learning Journal*, 2(55):175–210, 2004.
15. A. Maedche and S. Staab. Ontology learning for the semantic web. *IEEE Intelligent systems*, 16(2):72–79, 2001.
16. J. Rabatel, S. Bringay, and P. Poncelet. Contextual sequential pattern mining. In *(ICDMW), ICDM*, pages 981–988. IEEE, 2010.
17. R. Srikant and R. Agrawal. Mining generalized association rules. In *VLDB*, pages 407–419, 1995.
18. G. Stumme, A. Hotho, and B. Berendt. Semantic web mining: State of the art and future directions. *J. Web Sem.*, 4(2):124–143, 2006.

Identifying Disputed Topics in the News

Orphee De Clercq[1,3], Sven Hertling[2], Veronique Hoste[1],
Simone Paolo Ponzetto[3], and Heiko Paulheim[3]

[1] LT3, Language and Translation Technology Team, Ghent University
{orphee.declercq,veronique.hoste}@ugent.be
[2] Knowledge Engineering Group, Technische Universität Darmstadt
hertling@ke.tu-darmstadt.de
[3] Research Group Data and Web Science, University of Mannheim
{simone,heiko}@informatik.uni-mannheim.de

Abstract. News articles often reflect an opinion or point of view, with certain topics evoking more diverse opinions than others. For analyzing and better understanding public discourses, identifying such contested topics constitutes an interesting research question. In this paper, we describe an approach that combines NLP techniques and background knowledge from DBpedia for finding disputed topics in news sites. To identify these topics, we annotate each article with DBpedia concepts, extract their categories, and compute a sentiment score in order to identify those categories revealing significant deviations in polarity across different media. We illustrate our approach in a qualitative evaluation on a sample of six popular British and American news sites.

Keywords: Linked Open Data, DBpedia, Sentiment Analysis, Online News

1 Introduction

The internet has changed the landscape of journalism, as well as the way readers consume news. With many newspapers providing a website available offering news for free, many people are no longer local readers who are subscribed to one particular newspaper, but receive news from many sources, covering a wide range of opinions. At the same time, the availability of online news sites allows for in-depth analysis of topics, their coverage, and the opinions about them. In this paper, we explore the possibilities of current basic Semantic Web and Natural Language Processing (NLP) technologies to identify topics carrying *disputed* opinions.

There are different scenarios in which identifying those disputed opinions is interesting. For example, media studies are concerned with analyzing the political polarity of media. Here, means for automatically identifying conflicting topics can help understanding the political bias of those sources. Furthermore, campaigns of paid journalism may be uncovered, e.g. if certain media have significant positive or negative deviations in articles mentioning certain politicians.

In this paper, we start with the assumption that DBpedia categories help us identify specific topics. Next, we look at how the semantic orientation of news articles, based on a lexicon-based sentiment analysis, helps us find disputed news. Finally, we apply our methodology to a web crawl of six popular news sites, which were analyzed for *both* topics and sentiment. To this end, we first annotate articles with DBpedia concepts, and then use the concepts' categories to assign topics to the articles. Disputed topics are located by first identifying significant deviations of a topics' average sentiment per news site from the news site's overall average sentiment, and selecting those topics which have both significant positive and negative deviations.

This work contributes an interesting application of combining Semantic Web and NLP techniques for a high-end task. The remainder of this paper is structured as follows: in the next section we describe related work (Section 2). Next, we present how we collected and processed the data used for our system (Section 3). We continue by describing some interesting findings of our approach together with some of its limitations (Section 4). We finish with some concluding remarks and prospects for future research (Section 5).

2 Background and Related Work

Text and data mining approaches are increasingly used in the social science field of media or content analysis. Using statistical learning algorithms, Fortuna et al. [6] focused on finding differences in American and Arab news reporting and revealed a bias in the choice of topics different newspapers report on or a different choice of terms when reporting on a given topic. Also the work by Segev and Miesch [17], which envisaged to detect biases when reporting on Israel, found that news reports are largely critical and negative towards Israel. More qualitative studies were performed, such as the discourse analysis by Pollak et al.[14] which revealed contrast patterns that provide evidence for ideological differences between local and international press coverage.These studies either focus on a particular event or topic [14,17] or use text classification in order to define topics [6], and most often require an upfront definition of topics and/or manually annotated training data. In this work, instead, we use semantic web technologies to semantically annotate newswire text, and develop a fully automatic pipeline to find disputed topics by employing sentiment analysis techniques.

Semantic annotation deals with enriching texts with pointers to knowledge bases and ontologies [16]. Previous work mostly focused on linking mentions of concepts and instances to either semantic lexicons like WordNet [5], or Wikipedia-based knowledge bases [7] like DBpedia [9]. DBpedia was for example used by [8] to automatically extract topic labels by linking the inherent topics of a text to concepts found in DBpedia and mining the resulting semantic topic graphs. They found that this is a better approach than using text-based methods. Sentiment analysis, on the other hand, deals with finding opinions in text. Most research has been performed on clearly opinionated texts such as product or movie reviews [15], instead of newspaper texts which are believed to be less opinionated.

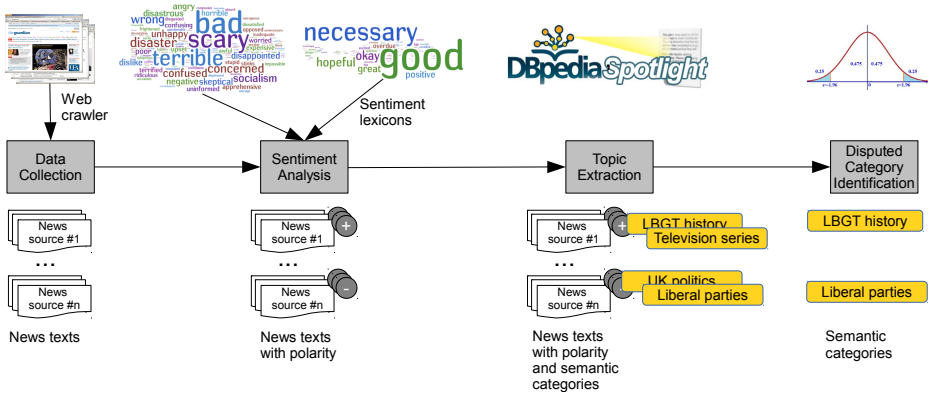


Fig. 1. An illustrative overview of our approach for identifying disputed topics.

An exception is the work performed by [2] in the framework of the European Media Monitor project [18].

While the combination of sentiment analysis and semantic annotation for the purpose discussed in this paper is relatively new, some applications have been produced in the past. The DiversiNews tool [20], for example, enables the analysis of text in a web-based environment for diversified topic extraction. Closely related are DisputeFinder [4] and OpinioNetIt [1]. The former is a browser extension which highlights known disputed claims and presents the user with a list of articles supporting a different point of view, the latter should allow to automatically derive a map of the opinions-people network from news and other web documents.

3 Approach

Our process comprises four steps, as depicted in Fig. 1. First, data is collected from online news sites. Next, the collected texts are augmented with sentiment scores and semantic categories, which are then used to identify disputed categories.

3.1 Data Collection

We have collected data from six online news sites. First, we looked at those having a high circulation and online presence. Another criterion for selection was the ability to crawl the website, since, e.g., dynamically loaded content is hard to crawl.

The six selected news sites fulfilling these requirements are shown in Table 1. We work with three UK and three US news sites. As far as the British news sites are concerned, we selected one rather conservative news site, the Daily Telegraph which is traditional right-wing; one news site, the Guardian, which

Table 1. Data Sources used in this paper. We report the news sites we used, the number of articles crawled, and the average article length in words.

Newspaper	Country	Website	# Art	Avg. artlen
The Daily Mirror	UK	http://www.mirror.co.uk/	1,024	422
The Daily Telegraph	UK	http://www.telegraph.co.uk	1,055	599
The Guardian	UK	http://www.guardian.co.uk/	1,138	638
The Huffington Post	US	http://www.huffingtonpost.com/	1,016	446
Las Vegas Review-Journal	US	http://www.reviewjournal.com/	1,016	618
NY Daily News	US	http://www.nydailynews.com/	1,016	338

can be situated more in the middle of the political spectrum though its main points of view are quite liberal; and finally also one tabloid news site, the Mirror, which can be regarded as a very populist, left-wing news site.[1] For the American news sites, both the Las Vegas Review–Journal and the Huffington Post can be perceived as more libertarian news sites[2], with the latter one being the most progressive [3], whereas the NY Daily News, which is also a tabloid, is still liberal but can be situated more in the center and is even conservative when it comes to matters such as immigration and crime.

The news site articles were collected with the python web crawling framework *Scrapy*[3]. This open-source software focuses on extracting items, in our case, news site articles. Each item has a title, an abstract, a full article text, a date, and an URL. We only crawled articles published in the period September 2013 – March 2014. Duplicates are detected and removed based on the article headlines.[4]

3.2 Sentiment Analysis

We consider the full article text as the context to determine the document's semantic orientation. The basis of our approach to define sentiment relies on word lists which are used to determine positive and negative words or phrases.

We employ three well-known sentiment lexicons. The first one is the Harvard General Inquirer lexicon – GenInq [19] – which contains 4,206 words with either a positive or negative polarity. The second one is the Multi-Perspective Question Answering Subjectivity lexicon – MPQA [22] – which contains 8,222 words rated between strong and weak positive or negative subjectivity and where morpho-syntactic categories (PoS) are also represented. The last one is the AFINN lexicon [12], which includes 2,477 words rated between -5 to 5 for polarity.

Before defining a news article's polarity, all texts were sentence-split, tokenized and part-of-speech tagged using the LeTs preprocessing toolkit [21]. In

[1] Cf. results of 2005 MORI research: http://www.theguardian.com/news/datablog/2009/oct/05/sun-labour-newspapers-support-elections.

[2] http://articles.latimes.com/2006/mar/08/entertainment/et-vegas8

[3] http://scrapy.org/

[4] The dataset and all other resources (e.g. RapidMiner processes) are made freely available to the research community at http://dws.informatik.uni-mannheim.de/en/research/identifying-disputed-topics-in-the-news.

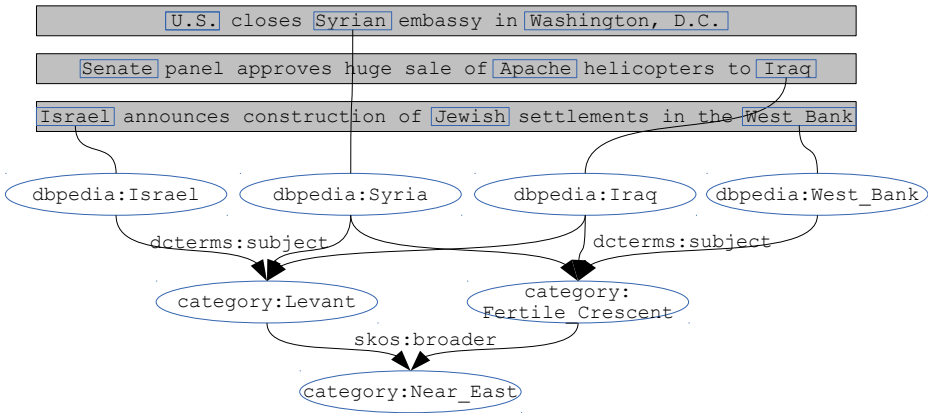


Fig. 2. Example of news texts annotated with DBpedia concepts, and categories extracted. The graph only shows a portion of the categories and their super category relationships.

a next step, various sentiment scores were calculated on the document level by performing a list look-up. For each document, we calculated the fraction of positive and negative words by normalizing over text length, using each lexicon separately. Then, in a final step we calculated the sum of the values of identified sentiment words, which resulted in an overall value for each document. That is, for each document d, our approach takes into consideration an overall lexicon score defined as:

$$lexscore(d) = \sum_{i=1}^{n} v_{w_i} \ . \tag{1}$$

where w_i is the i-th word from d matched in the lexicon at hand, and v_{w_i} its positive or negative sentiment value.

3.3 Topic Extraction

We automatically identify the topics of our news articles on the basis of a two-step process. First, we identify concepts in *DBpedia* [9]. To that end, each article's headline and abstract are processed with *DBpedia Spotlight* [10]. Next, categories for each concept are created, corresponding to the categories in Wikipedia: we extract all direct categories for each concept, and add the more general categories two levels up in the hierarchy.

These two phases comprise a number of *generalizations* to assign topics to a text. First, processing with DBpedia Spotlight generalizes different *surface forms* of a concept to a general representation of that concept, e.g. *Lebanon*, *Liban*, etc., as well as their inflected forms, are generalized to the concept `dbpedia:Lebanon`. Second, different DBpedia concepts (such as `dbpedia:Lebanon`, `dbpedia:Syria`) are generalized to a common category (e.g. `category:Levant`). Third, categories (e.g. `category:Levant`, `category:Fertile_Crescent`) are generalized to super

Table 2. Number of concepts identified by DBpedia Spotlight, and DBpedia categories extracted by source. The totals represent the number of unique concepts and categories.

Newspaper	# Concepts	# Categories
The Daily Mirror	971	10,017
The Daily Telegraph	784	8,556
The Guardian	605	6,919
The Huffington Post	400	6,592
Las Vegas Review-Journal	227	2,761
NY Daily News	942	10,540
Total	2,825	22,821

categories (e.g. `category: Near_East`). We provide an illustration of this generalization process in Fig. 2.

The whole process of topic extraction, comprising the annotation with DBpedia Spotlight and the extraction of categories, is performed in the *RapidMiner Linked Open Data Extension* [13]. Table 2 depicts the number of concepts and categories extracted per source. It can be observed that the number of categories is about a factor of 10 larger than the number of concepts found by DBpedia Spotlight alone. This shows that it is more likely that two related articles are found by a common category, rather than a common concept.

3.4 Disputed Categories Extraction

We identify disputed categories (and hence, topics) as follows:

1. First, for each news site we produce a global sentiment-orientation profile based on the overall sentiment scores (Equation 1): this is meant to model the coarse sentiment bias of a specific news source and avoid effects occurring due to the typical vocabulary of a news source.
2. Next, we identify those DBpedia categories for which the sentiment score deviates significantly from the global sentiment score. Since these follow a Gaussian distribution (Cf., Fig. 3, we can apply a z-test. From the overall number of texts n collected from a news site, the mean μ and standard deviation σ of the sentiment scores are computed , together with the average sentiment $M(c)$ of each category c. The latter is computed as

$$M(c) = \frac{1}{|C|} \sum_{d \in C} lexscore(d), \tag{2}$$

where C is the set of all articles annotated with category c, and $lexscore(d)$ is one of our three lexicon-based scoring functions. We can then compute the category's z score as:

$$z(c) = \frac{M(c) - \mu}{\frac{\sigma}{n}} \tag{3}$$

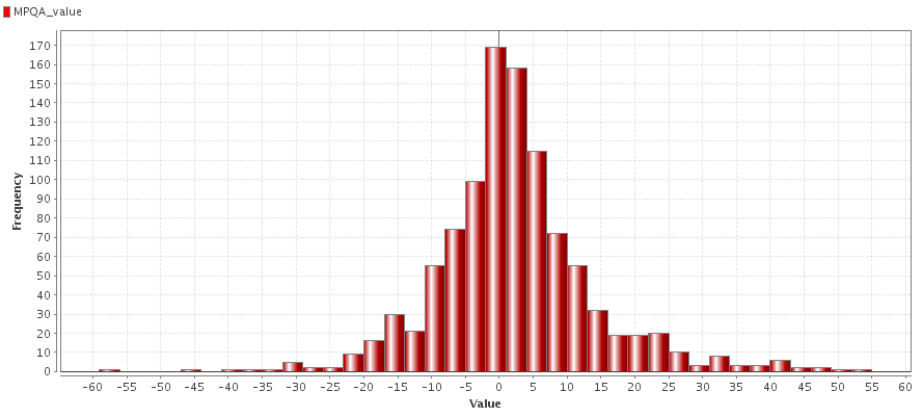


Fig. 3. Example distribution of the sentiment scores. The histogram shows the distribution of the MPQA sentiment score for texts from The Guardian.

If the z score is positive, articles in the category c are more positive than the average of the news source and the other way around. By looking up that z score in a Gaussian distribution table, we can discard those deviations that are statistically insignificant. For instance, the Mirror contains three articles annotated with the category `Church of Scotland`, with an average AFINN sentiment score of 20.667, which is significant at a z-value of 2.270.

3. In the last step, we select those categories for which there is at least one significant positive and one significant negative deviation. If two disputed categories share the same extension of articles (i.e. the same set of articles is annotated with both categories), we merge them into a cluster of disputed categories.

4 Analysis

The output of our system is presented in Table 3, showing that up to 19 disputed topics can be identified in our sample. In what follows we present some interesting findings based on a manual analysis of the output and we also draw attention to some limitations of our current approach. In general, we opt in this work for a validation study of the system output – as opposed, for instance, to a gold-standard based evaluation. This is because, due to the very specific nature of our problem domain, any ground truth would be temporally bound to a set of disputed topics for a specific time span.

4.1 Findings

If we look at the different percentages indicating the amount of articles found with a significant positive or negative sentiment, we see that these numbers differ

Table 3. Number of categories with a significant positive and/or negative sentiment per source, percentage of number of topics covered using the different lexicons, and the total number of disputed topics.

	GenInq			MPQA			AFINN		
Newspaper	pos	neg	%	pos	neg	%	pos	neg	%
The Daily Mirror	197	362	54.59	0	0	0	103	120	21.78
The Daily Telegraph	268	99	34.79	0	0	0	185	138	30.62
The Guardian	152	455	53.34	389	154	47.72	176	266	38.84
The Huffington Post	165	159	31.89	285	48	32.78	140	95	23.13
Las Vegas Review-Journal	70	75	14.27	54	68	12.01	92	68	15.75
NY Daily News	329	192	51.28	150	270	41.34	305	223	51.97
Disputed	19			11			17		

among the lexicons. The Daily Mirror seems to contain most subjective articles when using the GenInq lexicon, a role played by The Guardian and The Daily News NY when using the MPQA lexicon and the AFINN lexicon, respectively. The largest proportions are found within the Daily Mirror and the NY Daily News, which is not surprising since these are the two tabloid news sites in our dataset. Though the Daily Telegraph and the Daily Mirror seem to have no significant deviations using the MPQA lexicon[5], we nevertheless find disputed topics among the other four news sites. Consequently, the MPQA has the fewest (11), followed by AFINN (17) and GenInq (19).

Initially, we manually went through the output list of disputed topics and selected two topics per lexicon that intuitively represent interesting news articles (Table 5). What draws the attention when looking at these categories is that these are all rather broad. However, if we have a closer look at the disputed articles we clearly notice that these actually do represent contested news items. Within the category `Alternative_medicine`, for example, we find that three articles focus on *medical marijuana legalization*. To illustrate, we present these articles with their headlines, the number of subjective words with some examples, and the overall GI lexicon value[6].

- *NY Daily News*. "Gov. Cuomo to allow limited use of medical marijuana in New York" → 7 positive (e.g. great, tremendous) and 5 negative (e.g. difficult, stark) words; GI value of 2.00.
- *NY Daily News*: "Gov. Cuomo says he won't legalize marijuana Colorado-style in New York", → 5 positive (e.g. allow, comfortable) and 8 negative (e.g. violation, controversial) words; GI value of -3.
- *Las Vegas Review*: "Unincorporated Clark County could house Southern Nevada medical marijuana dispensaries", → 26 positive (e.g. ensure, accommodate) and 10 negative (e.g. pessimism, prohibit) words; GI value of 16.

[5] This might be due to MPQA's specific nature, it has different gradations of sentiment and also PoS tags need to be assigned in order to use it

[6] However, as previously mentioned in Section 3, for the actual sentiment analysis we only considered the actual news article and not its headline or abstract.

Table 4. Article and sentiment statistics of two categories for each lexicon

	Category/topic	# articles	# UK	# US	# pos	# neg	# neut
GenInq	`Alternative_medicine`	5	3	2	3	2	0
	`Death`	13	11	2	5	7	1
MPQA	`Government_agencies`	13	7	6	7	5	1
	`LGBT_history`	11	5	6	7	4	0
AFINN	`Democratic_rights`	7	4	3	4	3	0
	`Liberal_parties`	34	5	29	26	7	1

Though the last article is clearly about a difficult issue within this whole discussion, we see that the Las Vegas Review-Journal reports mostly positive about this subject which could be explained by its libertarian background. Whereas the NY Daily News, which is more conservative regarding such topics, reports on this positive evolution by using less outspoken positive and even negative language. A similar trend is reflected in the same two news sites when reporting on another contested topic, i.e. *gay marriage*, which turns up using the MPQA lexicon in the category `LGBT_history`. We again present some examples.

- *Las Vegas Review*: "Nevada AG candidates split on gay marriage" → 25 positive: 16 weak (allow, defense) and 9 (clearly, opportunity) are strong subjective and 13 negative: 10 weak (against, absence) and 3 (heavily, violate) strong subjective. MPQA value of 19.
- *NY Daily News*: "Michigan gov. says state won't recognize same-sex marriages", → 7 positive: 5 weak (reasonable, successfully) and 2 strong (extraordinary, hopeful) subjective and 9 negative: 4 weak (little, least) and 5 strong subjective (naive, furious). MPQA value of -5.

Another interesting finding we discover is that for four out of six categories, the articles are quite evenly distributed between UK and US news sites and that two categories stand out: `Death` seems to be more British and `Liberal_parties` more American. If we have a closer look at the actual articles representing these categories we see 9 out of the 11 `Death` articles actually deal with murder and were written for the Daily Mirror which is a tabloid news site focusing more on sensation. As far as the 34 American articles regarding liberal parties are concerned, we notice that all but six were published by the Las Vegas Review-Journal which is known for its libertarian editorial stance.

These findings reveal that using a basic approach based on DBpedia category linking and lexicon-based sentiment analysis already allows us to find some interesting, contested news articles. Of course, we are aware that our samples are too small to make generalizing assumptions which brings us to a discussion of some of the limitations of our current approach.

4.2 Limitations

In order to critically evaluate the limitations of our approach, we first had a look at the actual "topic representation". Since we use the lexicons as a basis

to find disputed topics, we randomly select 20 news articles that show up under a specific category per lexicon and assess its representativeness. We found that, because of errors in the semantic annotation process, out of these 60 examples, only 34 were actually representative of the topic or category in which they were represented. If we look at the exact numbers per lexicons, this amounts to an accuracy of 55% in the GenInq, one of 70% in MPQA and one of 40% in AFINN. Examples of mismatches, i.e. where a DBpedia Spotlight concept was misleadingly or erroneously tagged, are presented next:

- AFINN, `category:Television_series_by_studio`, tagged concepts: United_States_Department_of_Veterans_Affairs, Nevada, ER_TV_series → article is about a poor emergency room, not about the TV series *ER*.
- GenInq, `category:Film_actresses_by_award`, tagged concepts: Prince_Harry_of_Wales, Angelina_Jolie → article is about charity fraud, Angelina Jolie is just a patron of the organization.

We performed the same analysis on our manually selected interesting topics (cf. Table 4) and found that actually 74 out of the 83 articles were representative.

When trying to evaluate the sentiment analysis we found that this is a difficult task when no gold standard annotations or clear guidelines are available. Various questions immediately come to mind: does the sentiment actually represent a journalist's or newspaper's belief or does it just tell something more about the topic at hand? For example, considering the news articles in the Guardian dealing with murder it might be that words such as "murder", "kill",... are actually included as subjective words within the lexicon. However, at the moment this latter question is overruled by our disputed topic filtering step, which discards topics that are negative across all news sites.

5 Conclusions and Future Work

In this paper, we have discussed an approach which finds disputed topics in news media. By assigning sentiment scores and semantic categories to a number of news articles, we can isolate those semantic categories whose sentiment scores deviate significantly across different news media. Our approach is entirely unsupervised, requiring neither an upfront definition of possible topics nor annotated training data. An experiment with articles from six UK and US news sites has shown that such deviations can be found for different topics, ranging from political parties to issues such as drug legislation and gay marriage.

There is room for improvement and further investigation in quite a few directions. Crucially, we have observed that the assignment of topics is not always perfect. There are different reasons for that. First, we annotate the whole abstract of an article and extract categories. Apart from the annotation tool (DBpedia Spotlight) not working 100% accurately, this means that categories extracted for minor entities have the same weight as those extracted for major ones. Performing keyphrase extraction in a preprocessing step (e.g. as proposed by Mihalcea and Csomai [11]) might help overcoming this problem.

In our approach, we only assign a global sentiment score to each article. A more fine-grained approach would assign different scores to individual entities found in the article. This would help, e.g. handling cases such as articles which mention politicians from different political parties. In that case, having a polarity value per entity would be more helpful than a global sentiment score. Furthermore, more sophisticated sentiment analysis combining the lexicon approach with machine learning techniques may improve the accuracy.

Our approach identifies many topics, some of which overlap and refer to a similar set of articles. To condense these sets of topics, we use categories' extensions, i.e. the sets of articles annotated with a category. Here, an approach exploiting both the extension as well as the subsumption hierarchy of categories might deliver better results. Another helpful clue for identifying media polarity is analyzing the *coverage* of certain topics. For example, campaigns of paid journalism can be detected by a news site having a few articles on products from a brand, which are not covered by other sites.

Although many issues remain open, we believe this provides a first seminal contribution that shows the substantial benefits of bringing together NLP and Semantic Web techniques for high-level, real-world applications focused on a better, semantically-driven understanding of Web resources such as online media.

Acknowledgements

The work presented in this paper has been partly funded by the PARIS project (IWT-SBO-Nr. 110067) and the German Science Foundation (DFG) project Mine@LOD (grant number PA 2373/1-1). Furthermore, Orphee De Clercq is supported by an exchange grant from the German Academic Exchange Service (DAAD STIBET scholarship program).

References

1. Rawia Awadallah, Maya Ramanath, and Gerhard Weikum. Opinionetit: Understanding the opinions-people network for politically controversial topics. In *Proceedings of CIKM '11*, pages 2481–2484, 2011.
2. Alexandra Balahur, Ralf Steinberger, Mijail Kabadjov, Vanni Zavarella, Erik van der Goot, Matina Halkia, Bruno Pouliquen, and Jenya Belyaeva. Sentiment analysis in the news. In *Proc. of LREC'10*, 2010.
3. Jon Bekken. Advocacy newspapers. In Christopher H. Sterling, editor, *Encyclopedia of Journalism*. SAGE Publications, 2009.
4. Rob Ennals, Beth Trushkowsky, John Mark Agosta, Tye Rattenbury, and Tad Hirsch. Highlighting disputed claims on the web. In *ACM International WWW Conference*, 2010.
5. Christiane Fellbaum, editor. *WordNet: An Electronic Lexical Database*. MIT Press, Cambridge, Mass., 1998.
6. Blaž Fortuna, Carolina Galleguillos, and Nello Cristianini. Detecting the bias in media with statistical learning methods. In *Text Mining: Theory and Applications*. Taylor and Francis Publisher, 2009.

7. Eduard Hovy, Roberto Navigli, and Simone Paolo Ponzetto. Collaboratively built semi-structured content and Artificial Intelligence: The story so far. *Artificial Intelligence*, 194:2–27, 2013.
8. Ioana Hulpus, Conor Hayes, Marcel Karnstedt, and Derek Greene. Unsupervised graph-based topic labelling using dbpedia. In *Proc. of WSDM '13*, pages 465–474, 2013.
9. Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N. Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick van Kleef, Sren Auer, and Christian Bizer. DBpedia – A Large-scale, Multilingual Knowledge Base Extracted from Wikipedia. *Semantic Web Journal*, 2013.
10. Pablo N. Mendes, Max Jakob, Andrés García-Silva, and Christian Bizer. Dbpedia spotlight: Shedding light on the web of documents. In *Proc. of the 7th International Conference on Semantic Systems (I-Semantics)*, 2011.
11. Rada Mihalcea and Andras Csomai. Wikify!: Linking documents to encyclopedic knowledge. In *Proc. of CIKM '07*, pages 233–242, 2007.
12. Finn Nielsen. A new ANEW: Evaluation of a word list for sentiment analysis in microblogs. In *Proc. of the ESWC2011 Workshop on Making Sense of Microposts: Big things come in small packages*, 2011.
13. Heiko Paulheim, Petar Ristoski, Evgeny Mitichkin, and Christian Bizer. Data mining with background knowledge from the web. In *RapidMiner World*, 2014. To appear.
14. Senja Pollak, Roel Coesemans, Walter Daelemans, and Nada Lavrač. Detecting contrast patterns in newspaper articles by combining discourse analysis and text mining. *Pragmatics*, 5:1947–1966, 2011.
15. Ana-Maria Popescu and Orena Etzioni. Extracting product features and opinions from reviews. In Anne Kao and Stephen R. Poteet, editors, *Natural Language Processing and Text Mining*, pages 9–28. Springer London, 2007.
16. Lawrence Reeve and Hyoil Han. Survey of semantic annotation platforms. In *Proc. of the 2005 ACM symposium on Applied computing*, pages 1634–1638. ACM, 2005.
17. Elad Segev and Regula Miesch. A systematic procedure for detecting news biases: The case of israel in european news sites. *International Journal of Communication*, 5:1947–1966, 2011.
18. Ralf Steinberger, Bruno Pouliquen, and Erik Van der Goot. An introduction to the europe media monitor family of applications. *CoRR*, abs/1309.5290, 2013.
19. Philip J. Stone, Dexter C. Dunphy, Marshall S. Smith, and Daniel M. Ogilvie. *The General Inquirer: A Computer Approach to Content Analysis*. MIT Press, Cambridge, MA, 1966.
20. Mitja Trampus, Flavio Fuart, Jan Bercic, Delia Rusu, Luka Stopar, and Tadej Stajner. Diversinews a stream-based, on-line service for diversified news. In *SiKDD 2013*, pages 184–188, 2013.
21. Marjan Van de Kauter, Geert Coorman, Els Lefever, Bart Desmet, Lieve Macken, and Veronique Hoste. Lets preprocess: The multilingual lt3 linguistic preprocessing toolkit. *Computational Linguistics in the Netherlands Journal*, 3:103–120, 2013.
22. Theresa Wilson, Janyce Wiebe, and Paul Hoffman. Recognizing contextual polarity in phrase-level sentiment analysis. In *Proc. of HLT05*, pages 347–354, 2005.

Lattice-Based Views over SPARQL Query Results

Mehwish Alam and Amedeo Napoli

LORIA (CNRS – Inria Nancy Grand Est – Université de Lorraine)
BP 239, Vandoeuvre-lès-Nancy, F-54506, France
{mehwish.alam,amedeo.napoli@loria.fr}

Abstract. SPARQL queries over semantic web data usually produce a huge list of tuples as answers that may be hard to understand and interpret. Accordingly, this paper focuses on Lattice Based View Access (LBVA), a framework based on Formal Concept Analysis, to provide a view using `View By` clause based on a concept lattice. This lattice can be navigated for retrieving or mining specific patterns in query results.

Keywords: Formal Concept Analysis, SPARQL Query Views, Lattice-Based Views.

1 Introduction

At present, the Web has become a potentially large repository of knowledge, which is becoming main stream for querying and extracting useful information. In particular, Linked Open Data (LOD) [1] provides a method for publishing structured data in the form of RDF. These RDF resources are interlinked with each other to form a cloud. SPARQL queries are used in order to make these resources usable, i.e., queried. Queries in natural language against standard search engines sometimes may require integration of data sources. The standard search engines will not be able to easily answer these queries, e.g., *Currencies of all G8 countries*. Such a query can be formalized as a SPARQL query over data sources present in LOD cloud through SPARQL endpoints for retrieving answers. Moreover, these queries may generate huge amount of results giving rise to the problem of information overload [4]. A typical example is given by the answers retrieved by search engines, which mix between several meanings of one keyword. In case of huge results, user will have to go through a lot of results to find the interesting ones, which can be overwhelming without any specific navigation tool. Same is the case with the answers obtained by SPARQL queries, which are huge in number and it may be harder for the user to extract the most interesting patterns. This problem of information overload raises new challenges for data access, information retrieval and knowledge discovery w.r.t web querying.

This paper proposes a new approach based on Formal Concept Analysis (FCA [5]). It describes a lattice-based classification of the results obtained by SPARQL queries by introducing a new clause "`View By`" in SPARQL query.

This framework, called Lattice Based View Access (LBVA), allows the classification of SPARQL query results into a concept lattice, referred to as a *view*, for data analysis, navigation, knowledge discovery and information retrieval purposes. The `View By` clause enhances the functionality of already existing `Group By` clause in SPARQL query by adding sophisticated classification and Knowledge Discovery aspects. Here after, we describe how a lattice-based view can be designed from a SPARQL query. Afterwards, a view is accessed for analysis and interpretation purposes which are totally supported by the concept lattice. In case of large data only a part of the lattice [8] can be considered for the analysis.

The intuition of classifying results obtained by SPARQL queries is inspired by web clustering engines [2] such as Carrot2[1]. The general idea behind web clustering engines is to group the results obtained by query posed by the user based on the different meanings of the terms related to a query. Such systems deal with unstructured textual data on web. By contrast, there are some studies conducted to deal with structured RDF data. In [4], the authors introduce a clause `Categorize By` to target the problem of managing large amounts of results obtained by conjunctive queries with the help of subsumption hierarchy present in the knowledge base. By contrast, the `View By` clause generates lattice-based views which provide a mathematically well-founded classification based on formal concepts and an associated concept lattice. Moreover, it also paves way for navigation or information retrieval by traversing the concept lattice and for data analysis by allowing the extraction of association rules from the lattice. Such data analysis operations allow discovery of new knowledge. Additionally, unlike `Categorize By`, `View By` can deal with data that has no schema (which is often the case with linked data). Moreover, `View By` has been evaluated over very large set of answers (roughly 100,000 results) obtained over real datasets. In case of larger number of answers, `Categorize By` does not provide any pruning mechanism while this paper describes how the views can be pruned using icerberg lattices.

The paper is structured as follows: Section 2 describes the motivation. Section 3 gives a brief introduction of the state of the art while Section 4 defines LBVA and gives the overall architecture of the framework. Section 5 discusses some experiments conducted using LBVA. Finally, Section 6 concludes the paper.

2 Motivation

In this section we introduce a motivating example focusing on why LOD should be queried and why the SPARQL query results need classification. Let us consider a query searching for museums where the exhibition of some artists is taking place along with their locations. A standard query engine is not adequate for answering such kind of questions as it will produce a separate list of museums with the artists whose work is displayed there and a separate list of museums

[1] http://project.carrot2.org/index.html

with their locations. However, a direct query over LOD will perform resource integration to provide answers to the query. This query generates a huge amount of results, which further needs manual work to group the interesting links.

Linked Open Data represents data as RDF (Resource Description Framework) graphs. An RDF graph is a set of RDF triples, i.e., $\langle subject, predicate, object \rangle$, which is represented as node-and-arc-labeled directed graphs. SPARQL[2] is the standard query language for querying RDF graphs, which is based on matching graph patterns against RDF graphs. According to the scenario described above, the SPARQL query is shown in Listing 1.1.

Listing 1.1: SPARQL Query Museum

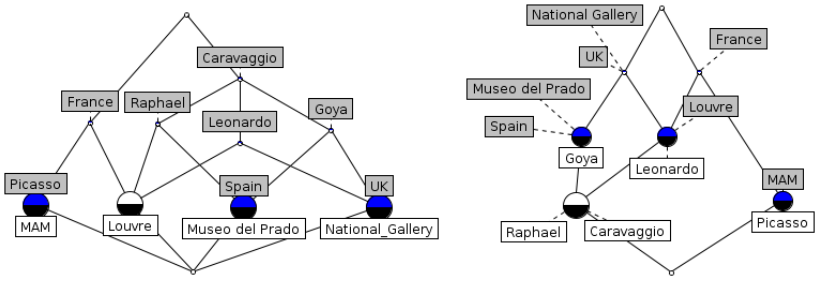
```
1 SELECT ?museum ?country ?artist WHERE {
2       ?museum rdf:type dbpedia-owl:Museum .
3       ?museum dbpedia-owl:location ?city .
4       ?city dbpedia-owl:country ?country .
5       ?painting dbpedia-owl:museum ?museum .
6       ?painting dbpprop:artist ?artist }
7       GROUP BY ?country ?artist
```

This query retrieves the list of museums along with the artists whose work is exhibited in a museum along with the location of a museum. `Lines 5 and 6` of this query retrieve information about the artists whose work is displayed in some museum. More precisely, the page containing the information on a museum (`?museum`) is connected to the page of the artists (`?artist`) through a page on the work of artist (`?painting`) displayed in the museum. An excerpt of the answers obtained by `Group by` clause is shown below:

```
Pablo_Picasso     Musee_d'Art_Moderne France
Leonardo_Da_Vinci Musee_du_Louvre     France
Raphael           Museo_del_Prado     Spain
```

The problem encountered while browsing such an answer is that there are thousands of results to navigate through. Even after using the `Group By` clause the answers are arranged into several small groups because first there is more than one grouping criteria and second, there are many values of the variables in the `Group By` clause. By contrast, the clause `View By` activates the LBVA framework, where a classification of the statements is obtained as a concept lattice (see Figure 1a). The concept lattice shown in Figure 1a is labeled in a reduced format (reduced labeling), meaning that if a parent class contains an attribute then this attribute is also inherited by its children concepts. Let us consider the parent concept has the attribute `France` then its two children concepts also have the attribute `France`. Now if the museums in `UK` displaying the work of `Goya` are to be retrieved, first the concept containing all the museums displaying the work of `Goya` is obtained and then the specific concept `Goya and UK` is retrieved by drilling down. Finally, the answer is `National Gallery`.

[2] http://www.w3.org/TR/rdf-sparql-query/



(a) Classes of Museums w.r.t Artists and Countries, e.g., the concept on the top left corner with the attribute `France` contains all the French Museums. (`VIEW BY ?museum`)

(b) Classes of Artists w.r.t Museums and Countries. (`VIEW BY ?artist`)

Fig. 1: Lattice Based Views w.r.t Museum's and Artist's Perspective .

3 Background

Formal Concept Analysis (FCA): FCA [5] is a mathematical framework used for a number of purposes, among which classification and data analysis, information retrieval and knowledge discovery [3]. Let G be a set of objects and M a set of attributes, and $I \subseteq G \times M$ a relation where gIm is true iff an object $g \in G$ has an attribute $m \in M$. The triple $\mathcal{K} = (G, M, I)$ is called a "formal context". Given $A \subseteq G$ and $B \subseteq M$, two derivation operators, both denoted by $'$, formalize the sharing of attributes for objects, and, in a dual way, the sharing of objects for attributes: $A' = \{m \in M \mid gIm \ for \ all \ g \in A\}$, $B' = \{g \in G \mid gIm \ for \ all \ m \in B\}$. The two derivation operators $'$ form a *Galois connection* between the powersets $\wp(G)$ and $\wp(M)$. Maximal sets of objects related to maximal set of attributes correspond to closed sets of the composition of both operators $'$ (denoted by $''$). Then a pair (A, B) is a formal concept iff $A' = B$ and $B' = A$. The set A is the "extent" and the set B is the "intent" of the formal concept (A, B). The set $\mathcal{C}_\mathcal{K}$ of all concepts from $\mathcal{K}$ is partially ordered by extent inclusion (or dually intent inclusion), denoted by $\leq_\mathcal{K}$ as $(A_1, B_1) \leq (A_2, B_2) \Leftrightarrow A_1 \subseteq A_2 (\Leftrightarrow B_2 \subseteq B_1)$. Consequently, $\mathcal{L}_\mathcal{K} = \langle \mathcal{C}_\mathcal{K}, \leq_\mathcal{K} \rangle$ forms the *concept lattice* of $\mathcal{K}$. There exist several algorithms [9] to build a concept lattice which also focus on efficiency of building the lattices for large number of objects.

In order to restrict the number of concepts in some cases iceberg concept lattices can be used [8], which contain only the top most part of the lattice. Formally, let $B \subseteq M$ and let minimum support, denoted by *minsupp*, be an integer representing a support threshold value. For a given concept (A,B), the support of B is the cardinality of A denoted by $|A|$. Relative support is given by $|A|/|G|$ and belongs to the interval $[0, 1]$. An intent B in concept (A,B) is said to be frequent as soon as $supp(B) = |A|/|G| \geq minsupp$. Likewise, a concept is called a frequent concept if its intent is frequent. The set of all frequent concepts of $\mathcal{K}$, for a given threshold, is called an iceberg concept lattice of $\mathcal{K}$. Along with iceberg lattices a stability index is also used for filtering the concepts. The stability index shows how much the concept intent depends on particular objects of the extent.

In some cases, a many-valued context is obtained instead of a formal context. A many-valued context is denoted by (G, M, W, I), where G is the set of objects, M is the set of attributes, W is the set of attribute values for each attribute and I represents a ternary relation between G, M, W, denoted as $I \subseteq G \times M \times W$. However, in order to obtain a one-valued binary context from the many valued context, scaling procedure is adopted. A scale S_m of an attribute m of a many-valued context is a one-valued context (G_m, M_m, I_m) with $m(G) = S_m$ for $m \in M$ and then the new set of attributes is $M_s = \bigcup_{m \in M} S_m$. During plain scaling the object set G remains unchanged, every many-valued attribute m is replaced by the scale attributes of scale S_m.

FCA also allows knowledge discovery using association rules. Duquenne-Guigues ($\mathcal{DG}$) basis for implications [6] is the minimal set of implications equivalent to the set of all valid implications for a formal context $\mathcal{K} = (G, M, I)$. An implication over the attribute set M in a formal context is of the form $B_1 \rightarrow B_2$, where $B_1, B_2 \subseteq M$. The implication holds iff every object in the context with an attribute in B_1 also has all the attributes in B_2. For example, when $(A_1, B_1) \leq (A_2, B_2)$ in the lattice, we have that $B_1 \rightarrow B_2$. $\mathcal{DG}$-basis represents all information lying in the concept lattice.

4 Lattice Based View Access

In this paper, we propose an approach called Lattice Based View Access for classification of SPARQL query results in the form of a concept lattice referred to as view. In the scenario of LOD, the RDF data and query processing procedure can not be controlled. Here we define views over RDF data by processing the set of tuples returned by the SPARQL query as answers.

SPARQL Queries with Classification Capabilities: The idea of introducing a `View By` clause is to provide classification of the results and add a knowledge discovery aspect to the results w.r.t the variables appearing in `View By` clause. Initially, the user poses SPARQL query of the form `SELECT` $?v_1$ $?v_2$ ... $?v_n$ `WHERE { condition/pattern } VIEW BY` $?v_l$. More formally, "$view\_by(v_l) : q(\overrightarrow{v})$" where $\overrightarrow{v}$ is a vector of variables containing free variables called answer variables and v_l is a variable in the `SELECT` clause of SPARQL query providing the viewing criteria. The evaluation of query q over RDF triple store generates answers in the form of tuples. A tuple is a vector of terms (set of constants) mapped to the answer variables in query q. The processing of a SPARQL query $q(\overrightarrow{v}) = q(v_1, v_2, \ldots, v_n)$ yields a set of tuples $R = \{(X_1^i, X_2^i, \ldots, X_n^i)\}$, where $i = \{1, \ldots, k\}$ where each tuple provides an elementary answer to the query $q(\overrightarrow{v})$.

Following the example in section 2, let us consider the user gives " `VIEW BY ?artist`" instead of `Group By` clause for the query in Listing 1.1. Then, $v_1 = artist$ is the object variable, $v_2 =$ `museum` and $v_3 =$ `country` are the attribute variables and Figure 1a shows the generated view. In Figure 1b, we have; $v_1 =$ `artist` is the object variable, $v_2 =$ `museum` and $v_3 =$ `country` are attribute variables.

Designing a Formal Context (G, M, W, I)***:*** The results obtained by the query are in the form of set of tuples, which are then organized as a many-valued context. Among the variables one variable appears in the `View By` clause and is considered as the *object variable*. All the other variables are considered as *attribute variables*. Let v_l be the object variable in $\overrightarrow{v} = (v_1, v_2, \ldots, v_n)$, X_l^i be the answers obtained for v_l, then attribute variables will be $v_1, \ldots, v_{l-1}, v_{l+1}, \ldots, v_n$. The answers associated to attribute variables can be given as $\{X_1^i, X_2^i, \ldots, X_{l-1}^i, X_{l+1}^i, \ldots, X_n^i\}$. Then, $G = \{X_l^i, i = 1, \ldots, k\}$ and M is the set of many valued attributes, given as $M = \{v_1, v_2, \ldots, v_{l-1}, v_{l+1}, \ldots, v_n\}$ and W represents the attribute values, i.e., $W = \{X_1^i, X_2^i, \ldots, X_{l-1}^i, X_{l+1}^i, \ldots, X_n^i\}$. The occurrence of an object and an attribute together in $R = \{(X_1^i, X_2^i, \ldots, X_n^i)\}$ gives the ternary relation I.

Let us continue the example discussed in section 2. The answers are organized into many-valued context as follows. The distinct values of the variable `?museum` are kept as a set of objects, so $G = \{MuseeduLouvre, MuseodelPrado\}$. The attribute variables `artist,country` provide the set of attributes $M = \{artist, country\}$ and the tuples related to the variables provide attribute values, $w_1 = \{Raphael, LeonardoDaVinci\}$ and $w_2 = \{France, Spain\ UK\}$. The obtained many-valued context is shown in Table 1. The corresponding nominally scaled one-valued context is shown in Table 2.

Museum	*Artist*	*Country*
Musee du Louvre	{Raphael, Leonardo Da Vinci, Caravaggio}	{France}
Musee d'Art Moderne	{Pablo Picasso}	{France}
Museo del Prado	{Raphael, Caravaggio, Francisco Goya}	{Spain}
National Gallery	{Leonardo Da Vinci, Caravaggio, Francisco Goya}	{UK}

Table 1: Many-Valued Context (Museum).

	Artist					*Country*		
Museum	Raphael	Da Vinci	Picasso	Caravaggio	Goya	France	Spain	UK
Musee du Louvre	×	×		×		×		
Musee d'Art Moderne			×			×		
Museo del Prado	×			×	×		×	
National Gallery		×		×	×			×

Table 2: One-Valued Context $\mathcal{K}_{Museum}$.

Building a Concept Lattice: Once the context is designed, a concept lattice (view) can be built using an FCA algorithm. This step is straight forward as soon as the context is provided. In the current study, we used *AddIntent*, which is an efficient implementation for building a concept lattice [9]. At the end of this step the concept lattice is built and the interpretation step can be considered. However, one limitation of the systems based on FCA is that they may encounter exponential time and space complexity in the worst case scenario for generating a concept lattice [7]. A view on SPARQL query in section 2, i.e, a concept lattice corresponding to Table 2 is shown in Figure 1a.

Interpretation Operations over a Lattice-Based Views: A formal context effectively takes into account the relations by keeping the inherent structure of

the relationships present in LOD as object-attribute relation. When a concept lattice is built, each concept keeps a group of terms sharing some attribute. This concept lattice can be navigated for searching and accessing particular LOD elements through the corresponding concepts within the lattice. This lattice can be drilled down from general to specific concepts or rolled up to find the general ones. For example, for retrieving the museums where there is an exhibition of `Caravaggio`'s paintings, it can be seen in the concept lattice shown in Figure 1a that the paintings of `Caravaggio` are displayed in `Musee du Louvre, Museo del Prado and National Gallery`. Now, in order to filter it by country, i.e., obtain `French` museums displaying `Caravaggio`. `Musee du Louvre` can be retrieved by navigating the same lattice. To retrieve museums located in `France` and `Spain`, a general concept containing all the French Museums with Caravaggio's painting is retrieved and then a specific concept containing the museums in `France` or `Spain` displaying `Caravaggio` can be accessed by navigation. The answer obtained will be `Musee du Louvre` and `Museo del Prado`.

After obtaining the view, i.e., the concept lattice, it can be accessed to obtain the $\mathcal{DG}$-basis of implications. For example, the rule `Goya, Raphael, Caravaggio` $\rightarrow$ `Spain` suggests that in Spain there exists a museum displaying the work of `Goya, Raphael, Caravaggio` (this implication is obtained from the view in Figure 1a). In order to get more specific answer, the user can browse through lattice and obtain `Museo Del Prado`.

5 Experimentation

5.1 DBpedia

DBpedia is currently comprised of a huge amount of RDF triples in many different languages which reflects the state of Wikipedia. Due to information extraction from crowd-sourced web site, triples present on DBpedia may contain incorrect information. Even if Wikipedia contains correct information, a parser may pick up wrong information [10]. Due to the above described reasons some of the properties may not be used uniformly. In the current experiment, we extracted the information about movies with their genre and location.

```
SELECT ?movie ?genre ?country WHERE {
?movie rdf:type dbpedia-owl:Film .
?movie dbpprop:genre ?genre .
?movie dbpprop:country ?country .}
VIEW BY ?movie
```

The obtained concept lattice contained 1395 concepts. Out of which 201 concepts on the first level were evaluated manually for correctness of the information about the movie genre. 141 concepts kept the genre information about the movie. 45% of these concepts contained wrong genre information as its intent (see first three concepts in Table 3). In such a case, the generated lattice-based view helps in separating music genre from the movie genre and further guide in introducing a new relation such as `soundtrackGenre` and adding

new triples to the knowledge base, for example, `dbpedia:The_Scorpion_King, dbpedia-owl:soundtrackGenre, dbpedia:Hard_Rock`.

ID	Supp.	Intent
C#1	17	`Hard Rock`
C#2	15	`Contemporary R&B`
C#3	18	`Jazz`
C#4	750	`United States`
C#5	1225	`India`
C#6	6	`France`

Table 3: Some Concepts from $\mathcal{L}_{DBpedia}$ (Concept Lattice for DBpedia)

Moreover, If we observe the obtained view, it can be seen that there are too few movies from countries other than United States and India. For example, C#4 and C#5 are the classes for movies from United States and India, where there are 1225 movies from India in DBpedia and 750 movies from United States. Finally, it can be concluded that the information present on DBpedia still needs to be corrected and completed.

The concept lattice can help in obtaining classes of movies w.r.t countries also. As this approach provides an added value to the already existing `Group By` clause, it is possible to find movies which are made in collaboration with several countries. For example, `The Scorpion King` was made in collaboration with `United States, Germany` and `Belgium`.

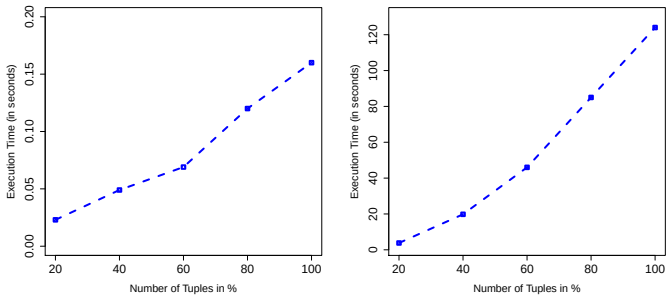
5.2 YAGO

The construction of YAGO ontology is based on the extraction of instances and hierarchical information from Wikipedia and Wordnet. In the current experiment, we posed a similar query to YAGO with the `View By` clause. While querying YAGO it was observed that the genre and location information was also given in the subsumption hierarchy of ontology. The first level of the obtained view kept the groups of movies with respect to their languages. e.g., the movies with genre `Spanish Language Films`. However, as we further drill down in the concept lattice we get more specific categories which include the values from the location variable such as `Spain, Argentina` and `Mexico`. Finally, it can be concluded that YAGO provides a clean categorization of movies by making use of the partially ordered relation between the concepts present in the concept lattice. YAGO also learns instances from Wikipedia and it contains many movies from all over the world. This observation gives the idea about the strong information extraction algorithm as it contains more complete information.

$\mathcal{DG}$-Basis of Implications for YAGO and DBpedia were computed. The implications were naively pruned with respect to support threshold. For DBpedia, the number of rules obtained were 64 for a support threshold of 0.11%. In case of YAGO, around 1000 rules were extracted on support threshold of 0.2%. Table 4 contains some of the implications obtained for both the datasets. It can be clearly observed that the support for the implications is much larger in YAGO

Impl. ID	Supp.	Implication
		YAGO
1.	96	`wikicategory RKO Pictures films` → `United States`
2.	46	`wikicategory Oriya language films` → `India`
		DBpedia
3.	3	`Historical fiction` → `United Kingdom@en`
4.	3	`Adventure fiction, Action fiction` → `Science fiction`

Table 4: Some implications from $\mathcal{DG}$-Basis of Implication (YAGO, DBpedia)



(a) Runtime for DBpedia (b) Runtime for YAGO

Fig. 2: Experimental Results.

than DBpedia, which points towards the completion of YAGO. This fact is actually useful in finding regularities in the SPARQL query answers which can not be discovered from the raw tuples obtained. For example, rule#1 states that `RKO picture films` is an American film production and distribution company as all the movies produced and distributed by them are from United States. Moreover, rule#2 says that all the movies in `Oriya language` are from `India`. Which actually points to the fact that Oriya is one of many languages that is spoken in India. On the other hand, some of the rules obtained from DBpedia are incorrect. For example, rule#3 states the strange fact that all the `historical fiction` movies are from `United Kingdom`. Same is the case with rule#4 which states that all the movies which are `Adventure fiction` and `Action fiction` are also `Science Fiction`, which may not actually be the case. Through the comparison of the $\mathcal{DG}$-Basis for both the datasets it can be observed that the YAGO may be more appropriate for further use by the application development tools and knowledge discovery purposes.

For each of the above queries we tested how our method scales with growing number of results. The number of answers obtained by DBpedia were around 4000 and the answers obtained by YAGO were 100,000. The experimental results of the runtime for the building the concept lattice are shown in Figure 2. Visualization of these experiments along with the implementation can be accessed online[3].

[3] `http://webloria.loria.fr/~alammehw/lbva/`

6 Conclusion and Discussion

In LBVA, we introduce a classification framework based on FCA for the set of tuples obtained as a result of SPARQL queries over LOD. In this way, a view is organized as a concept lattice built through the use of `View By` clause that can be navigated where information retrieval and knowledge discovery can be performed. Several experiments show that LBVA is rather tractable and can be applied to large data. For future work, we intend to use pattern structures with a graph description for each considered object, where the graph is the set of all triples accessible w.r.t reference object.

References

1. Christian Bizer, Tom Heath, and Tim Berners-Lee. Linked data - the story so far. *Int. J. Semantic Web Inf. Syst.*, 5(3):1–22, 2009.
2. Claudio Carpineto, Stanislaw Osiński, Giovanni Romano, and Dawid Weiss. A survey of web clustering engines. *ACM Comput. Surv.*, 41(3):17:1–17:38, July 2009.
3. Claudio Carpineto and Giovanni Romano. *Concept data analysis - theory and applications.* Wiley, 2005.
4. Claudia d'Amato, Nicola Fanizzi, and Agnieszka Lawrynowicz. Categorize by: Deductive aggregation of semantic web query results. In Lora Aroyo, Grigoris Antoniou, Eero Hyvönen, Annette ten Teije, Heiner Stuckenschmidt, Liliana Cabral, and Tania Tudorache, editors, *ESWC (1)*, volume 6088 of *Lecture Notes in Computer Science*, pages 91–105. Springer, 2010.
5. Bernhard Ganter and Rudolf Wille. *Formal Concept Analysis: Mathematical Foundations.* Springer, Berlin/Heidelberg, 1999.
6. J.-L. Guigues and V. Duquenne. Familles minimales d'implications informatives résultant d'un tableau de données binaires. *Mathématiques et Sciences Humaines*, 95:5–18, 1986.
7. Sergei O. Kuznetsov and Sergei A. Obiedkov. Comparing performance of algorithms for generating concept lattices. *J. Exp. Theor. Artif. Intell.*, 14(2-3):189–216, 2002.
8. Gerd Stumme, Rafik Taouil, Yves Bastide, and Lotfi Lakhal. Conceptual clustering with iceberg concept lattices. In R. Klinkenberg, S. Rüping, A. Fick, N. Henze, C. Herzog, R. Molitor, and O. Schröder, editors, *Proc. GI-Fachgruppentreffen Maschinelles Lernen (FGML'01)*, Universität Dortmund 763, October 2001.
9. Dean van der Merwe, Sergei A. Obiedkov, and Derrick G. Kourie. Addintent: A new incremental algorithm for constructing concept lattices. In Peter W. Eklund, editor, *ICFCA*, volume 2961 of *Lecture Notes in Computer Science*, pages 372–385. Springer, 2004.
10. Dominik Wienand and Heiko Paulheim. Detecting incorrect numerical data in dbpedia. In Valentina Presutti, Claudia d'Amato, Fabien Gandon, Mathieu d'Aquin, Steffen Staab, and Anna Tordai, editors, *ESWC*, volume 8465 of *Lecture Notes in Computer Science*, pages 504–518. Springer, 2014.

Visual Analysis of a Research Group's Performance thanks to Linked Open Data

Oscar Peña, Jon Lázaro, Aitor Almeida, Pablo Orduña, Unai Aguilera, Diego López-de-Ipiña

Deusto Institute of Technology - DeustoTech, University of Deusto
Avda. Universidades 24, 48007, Bilbao, Spain
{oscar.pena, jlazaro, aitor.almeida, pablo.orduna, unai.aguilera, dipina}@deusto.es

Abstract. Managing data within a research unit is not a trivial task due to the high number of entities to deal with: projects, researchers, publications, attended events, etc. When all these data are exposed on a public website, the need to have it updated is fundamental to avoid getting an incorrect impression of the group's performance. As research centres websites are usually quite static, external documents are generated by managers, resulting in data redundancy and out-of-date records. In this paper, we show our efforts to manage all these data using Labman, a web framework that deals with all the data, links entities and publishes them as Linked Open Data, allowing to get insightful information about the group's productivity using visual analytics and interactive charts.

1 Introduction

Managing metadata effectively within a research unit is an ambitious goal, as information systems need to deal with the relationships among the entities that form the organization's data model: projects, publications, researchers and project managers, topics, etc. Most research groups expose their data using a well known Content Management System (CMS) such as Joomla![1], WordPress[2] or Drupal[3]. Nonetheless, in order to extract valuable knowledge from all those data, external tools are needed to perform data analysis techniques. Exporting data in easy to handle formats from the CMS's databases usually leads to the creation of external documents which store data that will later be analysed.

This common situation has the following drawbacks: external documents (e.g., CSV, spreadsheets, text files, etc.) cause data redundancy, resulting in data quality, completeness and updating issues. This gets worse when investigators have their own personal pages (outside the system) where they show the achievements of their researching careers, funding data is managed by the accounting department and so on. When data needs to be updated in different

[1] http://joomla.org/
[2] http://wordpress.com/
[3] http://drupal.org/

systems, the expected outcome is that at some point data is going to be outdated somewhere, thus leading to errors when trying to get the whole picture of a research unit's performance.

Therefore, we present our efforts towards managing our research group's data, avoiding redundancy, improving quality and sharing the data in a standardized and interoperable way. Labman (Laboratory Management) is presented as a tool to manage all these data, publishing them as Linked Open Data. Linked Data allows to uniquely identify each entity instance with an URI, encouraging the creation of relationships among instances in order to discover patterns and insights in a dataset. Labman is a web application developed in Python using Django[4], and is Open Sourced on its Github's repository page[5], where it can be downloaded and contributed to. Labman is developed to substitute a previous Joomla! plugin developed at the research unit to publish publication data as RDF [1], thus overtaking the previously mentioned limitations.

This paper is structure as follows: First, we discuss similar efforts in section 2. Next, section 3 elaborates on the benefits of publishing information as Linked Data. Section 4 exhibits how patterns and knowledge can be extracted thanks to visualization techniques. Finally, conclusions and future work are addressed in section 5.

2 Related work

Even though some plugins have been developed to publish data stored within CMS systems as RDF (Resource Description Framework) files and RDFa metadata[6], they lack the ability to both make it accesible through a SPARQL endpoint (not allowing complex queries from external entities) and the advantages of publishing them following the Linked Data principles.

Research metadata visualization has also been studied by works such as [2] and [3], where authors use techniques from the visual analytics area to extract insights of research evolution in the studied cases. However, these works do not take the interlinking advantages of semantic descriptions, working with static dumps of database data.

The efforts of iMinds Multimedia Lab [4] demonstrates the potential insights that visual analytics provide when analysing research status on a country-level basis (i.e., applied to the whole research system of Belgium), publishing more than 400 million triples. Whereas users can get a full picture of the nation's research status, it does not substitute the information systems of the individual research centres.

ResearchGate[7] is a social networking site for scientist and researchers to share their work, providing metrics ands statistics to show their performance.

[4] https://djangoproject.com/
[5] https://github.com/OscarPDR/labman_ud
[6] http://rdfa.info/
[7] http://www.researchgate.net/

The focus is set on individuals to promote their work, whereas our proposal focuses on providing information on a research unit level basis.

Linked Universities, according to the definition on their website[8] "*is an alliance of european universities engaged into exposing their public data as linked data*". Specially focused on sharing educational data (e.g., courses, educational materials, teachers information, etc.), it also promotes the publishing of research and publication-related data. Linked Universities highlights the needs to have common shared vocabularies and tools to allow interoperability among how people access information about different institutions. Labman takes the recommendations from this alliance at its own core to avoid loosing the benefits provided by shared standards and vocabularies.

Finally, VIVO[9] is a huge project that provides an Open Source semantic web application to enable the discovery of researchers across institutions. VIVO allows any university to manage their data, and publish it using the VIVO ontology. VIVO is specially used among American universities.

3 Publication of resources as Linked Open Data

Linked Data (LD) is a series of principles and best practices to publish data in a structured way, encouraged by Tim Berners-Lee and the W3C [5]. LD is built over web standards such as HTTP, RDF and URIs, in order to provide information in a machine readable format. Every resource has its own URI, becoming a unique identifier for the data entity through all the system, thus avoiding data redundancy. Should somebody decide to extend the description of a given resource in its own dataset, both resources can be linked using the *rdfs:seeAlso* property, addressing that both resources refer to the same conceptual entity. The use of *rdfs:seeAlso* over *owl:sameAs* is preferred due to the semantic meaning difference between these properties: the former links two resources which refer to the same entity (maybe through different vocabularies), whereas the later connects two resources described in quite a similar way in different datasets.

The implicit linkage between resources in LD also allows to interconnect resources among them, e.g., a research project with the descriptions of people working on it and the related articles published as the outcomes of the study. Although this feature can also be achieved through plain relational database models, LD allows to connect references to external datasets, so complex queries can be performed in SPARQL, avoiding the potential headaches of joining consecutive SQL sentences and the need of having all the data in our system.

The "*Open*" term in Linked Open Data indicates that is freely available to everyone to use and republish data as they wish, without copyright and patent restrictions. All the information published on Labman is of public domain by default, making it freely consumable through its SPARQL endpoint. However, there is an option to mark a certain's project funding as *private*. If marked, this financial information will be used for the generation of top level visualizations

[8] `http://linkeduniversities.org/`
[9] `http://www.vivoweb.org/`

(those which give a full view of the unit's performance), but no funding charts will be rendered for that specific project and the funding amounts triples will not be generated.

3.1 Managing data within Labman

To encourage the adoption of Labman among the Semantic Web community, we have used well known vocabularies to describe the data of the different entities in our data model. The Semantic Web for Research Communities (SWRC) [6] ontology has been extended to provide financial information about research projects, together with some missing properties to link resources in our model. SWRC-FE (SWRC Funding Extension) is available for any semantic enthusiast to be used in their descriptions[10]. Researchers are mainly described using FOAF[11], while publications are defined thanks to the complete BIBO ontology[12]. Actually research topics are published using the Modular Unified Tagging Ontology (MUTO)[13], but we are considering to reference external topic datasets in the near future.

Labman stores data both in a relational database and as RDF triples (the relational database is used to increase performance and to allow non-semantic erudits to work with relational dumps). When an instance of any model is saved in Labman, a call is triggered to publish the instance and its attributes as RDF, generating or updating the referenced resource and its associated triples thanks to the rules of mapping specified for each model. Those triples are loaded into an Open Link Virtuoso[14] instance to be later on accessible through the dedicated SPARQL endpoint[15]. Semantics can be enabled/disabled on demand for a full deploy of Labman through general settings (useful when installing a local instance of Labman to get a taste of the system and easing the transition from a legacy relational database model). A single management command allows to make a full dump of the relational database and publish it as RDF triples. The list of available extra commands within labman can be consulted through the *–help* modifier of Django's *manage.py* command line feature.

To help with publications data adquisition, a Zotero[16] parser has been developed in order to extract all publication-related data and import it in Labman's system, publishing it as Linked Open Data using the previous described ontologies. Thanks to Zotero and the browser plugins, all metadata regarding a publication is extracted from well known publication indexing databases such as Web of Science, CiteSeer, Springer, Elsevier and so forth.

As the same authors may appear under slightly different names on different sites, Labman implements an author alias algorithm to perform term disam-

[10] `http://www.morelab.deusto.es/ontologies/swrcfe`
[11] `http://www.foaf-project.org/`
[12] `http://bibliontology.com/`
[13] `http://muto.socialtagging.org/core/v1.html`
[14] `http://virtuoso.openlinksw.com/`
[15] `http://www.morelab.deusto.es/labman/sparql`
[16] `https://www.zotero.org/`

biguation and apply the corresponding substitutions. This simple algorithm uses Python's difflib library[17] to compare strings (e.g., author full names), taking as input string pairs of all the author names present in Labman, and returning a similarity ratio between them (as shown in table 1. If the ratio is greater than the given threshold, both strings are sent to Labman's administrators for dissambiguation checking. If the match is approved, the *incorrect* author name from the pair is assigned as an alias of the valid name. A periodic background task unlinks all the referenced triples to the invalid alias, and assigns them to the correct author resource.

Table 1. Character sequence similarity

Sequence #1	Sequence #2	Similarity ratio
Diego López de Ipiña	D. Lopez-de-Ipiña	0.700
E. Fernández	F. Hernández	0.879
Oscar Pena	Oscar Peña del Rio	0.621

4 Understanding research data through visualizations

The adage "*A picture is worth a thousand words*" fits perfectly when visualizing research related info, as the huge amounts of data related to projects, funding agencies, organizations, researchers and so forth makes them perfect candidates to be rendered using visual representations, instead of displaying all the information in text form without highlighting underlying connections.

In order to access and interact with the visualizations through any web browser, web graphics libraries such as Google Charts[18], d3js[19] and sigma.js[20] have been used. Charts and graphs are rendered on the screen using JavaScript, with data extracted from Labman using Python.

Due to the interlinked nature of Linked Open Data, most visualizations showing linked entities are rendered as graphs and linked nodes. Visualizations are available on the *Charts* section[21] and on the extended information subsections of the webpage.

4.1 Funding

Project managers and principal investigators usually depend on funds to continue their research work. In a transparency effort, Labman allows to consult how much money is gathered from public entities, as displayed in figure 1.

[17] `https://docs.python.org/2/library/difflib.html`
[18] `https://developers.google.com/chart/`
[19] `http://d3js.org/`
[20] `http://sigmajs.org/`
[21] `http://www.morelab.deusto.es/labman/charts/`

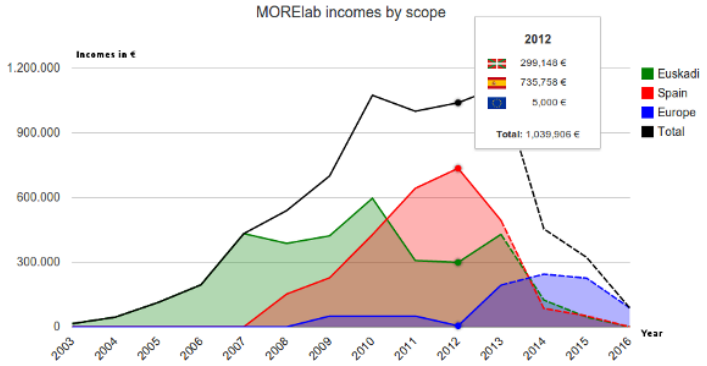


Fig. 1. Gathered funding timeline, by geographical scope.

Funds are provided by public administrations and organizations, usually under a named funding call. Labman also takes this information into account and allows to compare different calls' performances. For example, the principal investigator can view historical records from european FP7 and spanish INNPACTO funding calls to design the new budget strategy for the forthcoming years. Geographical scopes can be defined and related to Geoname's[22] feature classes, to classify funding call *levels* according to their effect area.

4.2 Project and publication collaborations

Research would not be possible without the collaborations of different researchers working together to generate new knowledge. Being able to detect research networks is a fundamental insight to have always present, together with the communities of practice our unit takes part in and the evolution and the interactions with members of external disciplines.

In figure 2, a force directed graph is selected to represent project collaborations present in the system. When hovering over an element, only the node's community is visible, allowing to consult who each person is related with. Node's size is calculated using Eigenvector centrality, a value which increases if the connection with other central nodes of the graph is relevant, and the color of each node indicates the community it belongs to. Community belonging is calculated using modularity, and a different color does not mean they do not work for the same organization, but that their connections make them beloging to a different group of interconnected people. The calculations for generating these graphs are further explained in [7]. Link weights take into account the strenght of the collaboration. For example, in projects, the more time spent working with a colleague, the stronger the connection will be, whereas the number of co-authored publications is a strong indicator of the preferences of publishing together. Collaboration edges create different triples, more accurate than the *foaf:knows* relation to analyse the relationship between two researchers.

[22] `http://www.geonames.org/`

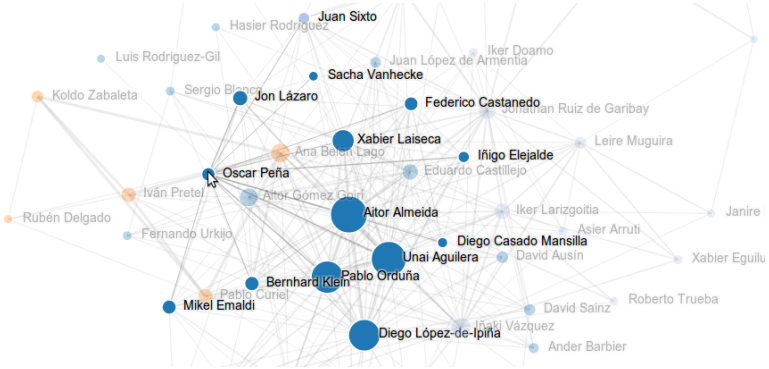


Fig. 2. Project collaborations of a researcher.

Figure 3 shows the egonetwork [8] of one of our researcher's (highlighted with a dotted circle), which represents the actual publication collaborations the researcher has with other members of the system. The stronger the links between two authors, the more publications they have produced together.

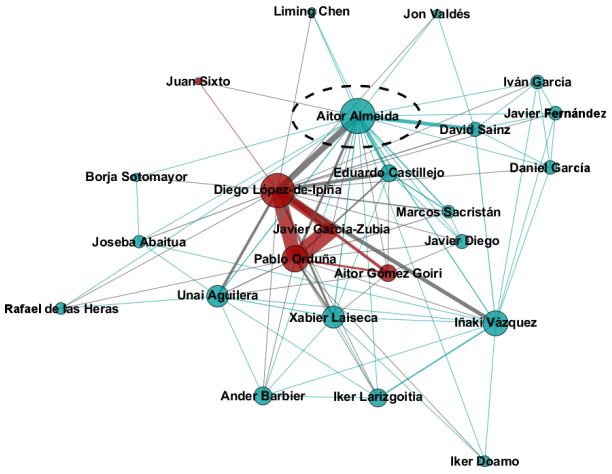


Fig. 3. Publications egonetwork of one of our researchers.

Sometimes relations between researchers are not explicit (e.g., two researchers have not worked together in a project or co-authored the same paper, but both of them work in the same knowledge area). In order to identify these relations we have implemented a similarity coefficient using the tags of the papers of each researcher. To ascertain the similarity of one researcher with another we have devised the following formula:

$$coef = \frac{|B \cap A|}{|A|}$$

Where A is the set of tags belonging to the base researcher and B is the set of tags belonging to the researcher which we want to compare the base researcher with. It must be taken into account that this similarity coefficient is not symmetrical. The reason is the topic similarity for a given researcher is considered within the whole of its tags, without taking into account the whole topics a related researcher works in. This situation is common for novel PhD students with a few published articles, who share all their topics with their advisors (due to their co-authorship), but senior researchers will have a broader set of areas they have worked on because of ther large research trajectory.

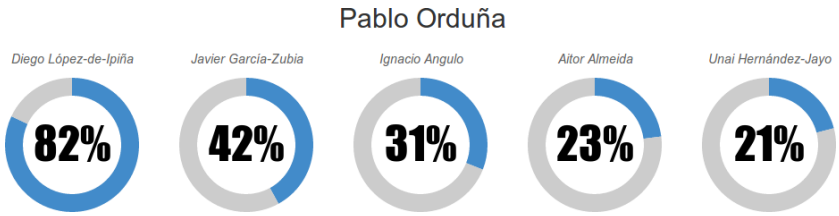


Fig. 4. Similar researchers to a given one using the previous normalized coeficient.

4.3 Topic insights

Together with the identification of research networks, knowing which topics those networks and the involved researchers are working in is fundamental to understand the most relevant areas the group is focusing on. Projects and publications are tagged with concepts in Labman, published as *dc:subject* triples using the *muto:Tag* ontology. The first obvious visual representation is to generate weighted lists (also known as tag or word clouds) of the topics used by a researcher. Figure 5 displays the topics used by one of our researchers, being the size of the tag representative of its weight (i.e., the bigger the tag, the more prolific in that area).

Research topic evolutions are also a good indicator to detect which areas the group is focused on. The historical evolution helps understanding which topics are no longer *hot* amongst researchers, and which topics have *died* to evolve into new research areas (e.g., from *Ubiquitous computing* to *Internet of Things* to *Weareable computing*). ArnetMiner [9] generates similar visualizations using automatically extracted metadata from the papers it finds for a researcher. However, many papers are not gathered, making those visualizations not to show the real status of the research.

Eventually, establishing a robust taxonomy of topics leads to the identification of interest groups and expertise hubs around topics, allowing to relate

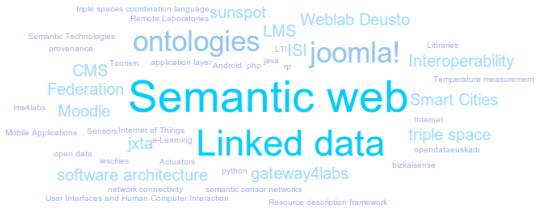


Fig. 5. Weighted list of the topics related to a person.

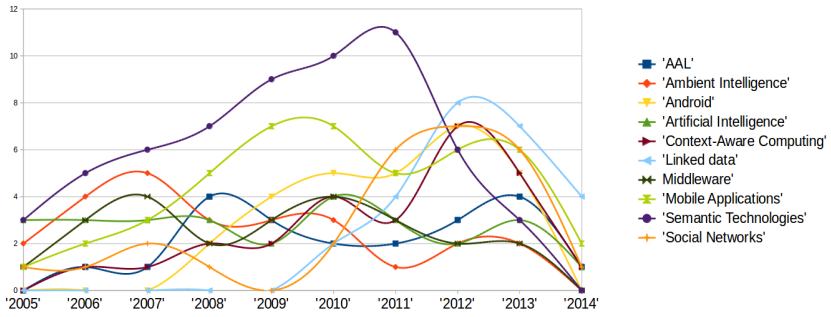


Fig. 6. Topic evolution of the research centre.

researchers, projects and publications automatically where no previous obvious hints were available to connect them. Describing resources using shared vocabularies and connecting to external knowledge datasets allows to create these links in a global space, opening new doors to knowledge discovery thanks to the use of Linked Data principles.

5 Conclusions and future work

Using dynamic visualizations of research information published as Linked Open Data, helps end-users (i.e., those consulting the information either from the research centre or visitors) to discover real connections between its members and the different entities modelled in the system. Visually representing which topics the researchers work on, who can be consulted about a certain area, and the historical collaborations within the group members can be of great help to guide the development of new project proposals, discover non-obvious potentials and address the key entities.

As future work, we will continue working on the generation of new visualizations to provide opportunities to improve the performance and strategic vision of a research centre. There is a strong continuous commitment to connect entities with external datasets, in order to evolve from a four-star to a full Linked Open Data data space, making Labman able to answer complex queries with data not present in our system, including other descriptions to the same resources

available on the Web of Data. Finally, a fine detail level when well defining and describing topics will allow for deeper analysis of data, taking into consideration the evolution of topics through time and how research areas are hierarquically structured. Actually, topics are cleaned and reviewed automatically on a regular basis to improve how resources are tagged. Better data completeness will lead to more enlightening reports, so automatizing even further the data adquisition stage will benefit all users.

6 Acknowledgements

The research activities described in this paper are partially funded by DeustoTech, Deusto Institute of Technology, a research institute within the University of Deusto and the Basque Government's Universities and Research department, under grant *PRE_2013_1_848*.

References

1. Mikel Emaldi, David Buján, and Diego López-de Ipina. Towards the integration of a research group website into the web of data. In *CAEPIA - Conferencia de la Asociación Española para la Inteligencia Artificial*, 2011.
2. Weimao Ke, Katy Borner, and Lalitha Viswanath. Major information visualization authors, papers and topics in the acm library. In *Information Visualization, 2004. INFOVIS 2004. IEEE Symposium on*, page r1–r1. IEEE, 2004.
3. Kevin W Boyack, Richard Klavans, and Katy Börner. Mapping the backbone of science. *Scientometrics*, 64(3):351–374, 2005.
4. Anastasia Dimou, Laurens De Vocht, Geert Van Grootel, Leen Van Campe, Jeroen Latour, Erik Mannens, and Rik Van de Walle. Visualizing the information of a linked open data enabled research information system. euroCRIS, May 2014. Delivered at the CRIS2014 Conference in Rome; to appear in the Procedia online CRIS2014 procs on ScienceDirect.
5. 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.
6. York Sure, Stephan Bloehdorn, Peter Haase, Jens Hartmann, and Daniel Oberle. The SWRC ontology – semantic web for research communities. In Carlos Bento, Amílcar Cardoso, and Gaël Dias, editors, *Progress in Artificial Intelligence*, number 3808 in Lecture Notes in Computer Science, pages 218–231. Springer Berlin Heidelberg, January 2005.
7. Vincent D. Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2008(10):P10008, October 2008. arXiv: 0803.0476.
8. Martin Everett and Stephen P. Borgatti. Ego network betweenness. *Social Networks*, 27(1):31–38, January 2005.
9. Jie Tang, Jing Zhang, Limin Yao, Juanzi Li, Li Zhang, and Zhong Su. Arnetminer: extraction and mining of academic social networks. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, page 990–998. ACM, 2008.

Machine Learning on Linked Data, a Position Paper

Peter Bloem and Gerben K. D. de Vries

System and Network Engineering Group
Informatics Institute, University of Amsterdam
uva@peterbloem.nl, g.k.d.devries@uva.nl

Abstract. The combination of linked data and machine learning is emerging as an interesting area of research. However, while both fields have seen an exponential growth in popularity in the past decade, their union has received relatively little attention. We suggest that the field is currently too complex and divergent to allow collaboration and to attract new researchers. What is needed is a simple perspective, based on unifying principles. Focusing solely on RDF, with all other semantic web technology as optional additions is an important first step. We hope that this view will provide a low-complexity outline of the field to entice new contributions, and to unify existing ones.

1 Introduction

Linked data is one of the most powerful frameworks for storing data, and machine learning (ML) is one of the most popular paradigms for data analysis, so it seems justified to ask what the union of the two has produced.[1] The answer is disappointing. Research papers, challenges [1], technical tools [2] and workshops [1, 3] exist, but for two such golden subjects, one would expect a significant proportion of machine learning research to deal with linked data by now.

In this paper we will focus on the lack of interest from the ML community: for ML researchers, the main impediment to getting involved with linked data is the complexity of the field. Researchers must learn about the Semantic Web, RDF, ontologies, data modeling, SPARQL and triple stores. Even if some subjects are not essential, it is difficult to see the forest for the trees. Besides the difficulty of understanding the Semantic Web, there is also the divergence of existing work, which ranges from tensor-based approaches on RDF graphs, to graph kernels on small RDF subgraphs, to relational learning techniques. A simple, unified view is hard to find.

2 A machine learning perspective

A common view of the intersection of machine learning and linked data is that machine learning can provide inference where traditional, logic-based methods

[1] We use "machine learning" as a catch-all term covering also data mining and knowledge discovery.

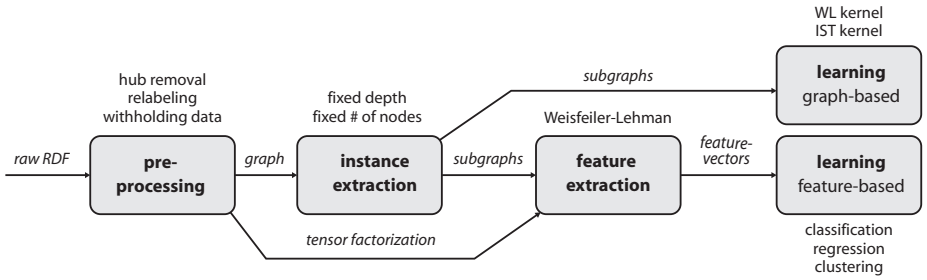


Fig. 1: An overview of a typical machine learning pipeline for RDF. Our aim here is not to provide a comprehensive framework, but to highlight some common steps.

fail [4], for instance to aid the effort of manually curating the Semantic Web [5]. We consider this the *Semantic Web perspective*. In contrast, we take a *machine learning perspective*: we see linked data as simply a new form of data.

In classical machine learning, the complexity and divergence of the field is controlled by what we will call the 'black-box principle'. Each machine learning method is expected to fit a simple mold: the input is a table of instances, described by several features with a target value to predict, and the output is a model predicting the target value.

The emergence of the semantic web upsets this view. In the semantic web, a dataset is no longer separated neatly into instances. It does not come with an obvious single learning task and target value, and the standard methods of evaluation do not fit perfectly. We require a new black box principle.

We think that the best way to unify machine learning and the Semantic Web is to *focus on RDF*. The Resource Description Framework (RDF) is the lowest layer in the Semantic Web stack. To understand it, we do not need to know about ontologies, reasoning and SPARQL. Of course, these can be important, but an ML researcher does not need to understand them to have the benefit.

A generic pipeline While the inside of the black box is up to the discretion of the researcher, it would help to have some standardized methods. We have drawn an example pipeline (Figure 1), to get from RDF to an ML model. We do not propose this as a catch-all pipeline (like a similar image in [6]), we simply expect that solving the most common tasks in machine learning from linked data[2] will often require one or more of these steps:

pre-processing RDF is a verbose data format, designed to store data for any future use. For machine learning purposes, it can help to reverse some of this verbosity [7]. Additionally, traditional methods like RDFS/OWL inferencing can be employed to create a graph that more efficiently exposes the relevant

[2] The most common tasks, from a SW perspective are probably *class prediction*, *property prediction* and *link prediction*. From the machine learning perspective these tasks can be regarded as classification, regression or ranking tasks. See [4].

information. In this step the researcher must also choose how to deal with constructs like blank nodes and reification.

instance extraction We assume that each of our instances is represented by a resource in an RDF graph.[3] However, the resource by itself contains no information. The actual description of the instances is represented by the neighborhood around the resource. Usually, a full subgraph is extracted to a given depth (e.g. [8, 9]), but more refined methods are likely possible.

feature extraction Most machine learning methods use feature vectors. Transforming RDF graphs to features, while retaining the subtleties of information contained in the RDF representation is probably the central problem in machine learning on RDF data.[4] The current state of the art for RDF is represented by the WL algorithm[5] [9] and tensor decomposition [11].

learning Once we have our feature vectors or graphs, we can feed them to a learner, to perform classification, regression or clustering.

Most graph kernels [8, 9] can be seen either as a graph learner or as a powerful feature extractor. The same holds for the RESCAL tensor factorization algorithm [11]. Other techniques, like Inductive Logic Programming (ILP), can be employed to solve a variety of RDF-based tasks [4][Section 3].

Evaluation In traditional machine learning, we can simply cut the table of instances and their features in two parts to obtain a training and test set. With RDF, the data is densely interconnected, and each prediction can change both the training and the test instances. Machine learning on RDF thus requires us to re-evaluate our standard evaluation approaches. We offer two guidelines:

Remove the target data from the whole dataset We recommend taking the value to be predicted and removing it from the dataset entirely, representing it as a separate table, mapping instances to their target values. This gives the researcher the certainty that they have not inadvertently left information from the test set in the training data. It can also speed up cross-validation, as the knowledge graph stays the same between folds [8, 9].

Refer to a real-world scenario Even when the target value is removed it can be complicated to judge whether additional information should be removed as well. If, for example, we are predicting a category for news articles, which has been inferred from more complex annotations by human experts, should we remove these annotations too? In such cases, it is best to refer back to the real world use case behind the learning task. In our example, we most likely want to replace the human annotators, so the scenario we want to model is one where their annotations are not available.

This gives us a rough picture of what a generic machine learning task might look like in the world of linked data. A dataset consists of a graph, with labeled

[3] There are exceptions, where each instance is represented by a specific relation, or by a particular subgraph. In such cases, the pipeline does not change significantly.

[4] In the field of relational learning this task is known as *propositionalization* [10]

[5] The WL algorithm is commonly presented as a graph kernel, but in its basic form it can also be seen as a feature extractor.

vertices and edges. In contrast to normal graph learning, however, the whole dataset is a single graph, with certain vertices representing the instances. If the task is supervised, a separate table provides a target value for each instance.

3 Outlook

We will finish with a sketch of what promises RDF holds, and what a community around machine learning on RDF might look like.

RDF as the standard data format in machine learning Currently, the most common way of sharing data in the ML community is in vector-based formats, for example most data in the UCI repository.[6] While the UCI repository has been of exceptional value to the community, this approach has several drawbacks: the semantic interpretation of the data is stored separately, the file formats may become out of date, and most importantly, the choices made in extracting the features cannot be reversed.

A better approach is to store the data in its most raw form. This means the data format should be independent of any intended use for the data, which is exactly what RDF is designed to do.

Competitions and Benchmark sets While there have been some machine learning challenges for RDF data, the uptake has so far been minimimal. We offer three guidelines for a good machine learning challenge on RDF. First, any challenge should contain only one aspect that is unusual in machine learning (ie. the data is represented as RDF). Everything else should be as conventional as possible. Ideally, the task boils down to binary classification with well-balanced classes. Second, the task should have a moving horizon: eg. the MNIST task [12] has seen its best error rate move down from 12% to 0.23% over 14 years. Finally an example script should be provided that performs the task. Both to give a starting point, and a target to aim for.

The linked data cloud as a single dataset The final part of our outlook for machine learning on linked data is a move away from single datasets. If our instance extraction algorithms crawl a dataset starting at the instance node and following relations to explore its neighborhood, it is a simple matter to let the extractor jump from one dataset to another by following the links already present. The machine learning researcher can remain ambivalent to which dataset she is working with: the instances will simply be subgraphs of the full linked data cloud.

4 Conclusion

Linked data is fast becoming one of the primary methods of exposing data for a wide range of institutions. The ML community should respond with a clear

[6] http://archive.ics.uci.edu/ml/datasets.html

package of methods and best practices to bring this type of data into the fold. What is needed, is a simple, lowest common denominator, a black box view for machine learning on RDF data, and a set of common techniques for data preprocessing.

We hope to start a conversation to unify our efforts, to lower the threshold for other machine learning researchers to join us, and to bring these communities closer together with a common language and a clear division of labor.

Acknowledgments This publication was supported by the Dutch national program COMMIT. We thank the reviewers for their valuable comments.

References

1. d'Amato, C., Berka, P., Svátek, V., Wecel, K., eds.: Proceedings of the International Workshop on Data Mining on Linked Data collocated with ECMLPKDD 2013. Volume 1082 of CEUR Workshop Proceedings. CEUR-WS.org (2013)
2. Paulheim, H., Fürnkranz, J.: Unsupervised generation of data mining features from linked open data. In Burdescu, D.D., Akerkar, R., Badica, C., eds.: WIMS, ACM (2012) 31
3. Paulheim, H., Svátek, V., eds.: Proceedings of the Third International Workshop on Knowledge Discovery and Data Mining Meets Linked Open Data. In Paulheim, H., Svátek, V., eds.: KNOW@LOD. (2014)
4. Rettinger, A., Lösch, U., Tresp, V., d'Amato, C., Fanizzi, N.: Mining the semantic web—statistical learning for next generation knowledge bases. Data Min. Knowl. Discov. **24**(3) (2012) 613–662
5. d'Amato, C., Fanizzi, N., Esposito, F.: Inductive learning for the semantic web: What does it buy? Semantic Web **1**(1-2) (2010) 53–59
6. Tresp, V., Bundschus, M., Rettinger, A., Huang, Y.: Towards machine learning on the semantic web. In: Uncertainty reasoning for the Semantic Web I. Springer (2008) 282–314
7. Bloem, P., Wibisono, A., de Vries, G.K.D.: Simplifying RDF data for graph-based machine learning. In: KNOW@LOD. (2014)
8. Lösch, U., Bloehdorn, S., Rettinger, A.: Graph kernels for RDF data. In Simperl, E., Cimiano, P., Polleres, A., Corcho, Ó., Presutti, V., eds.: ESWC. Volume 7295 of Lecture Notes in Computer Science., Springer (2012) 134–148
9. de Vries, G.K.D.: A fast approximation of the Weisfeiler-Lehman graph kernel for RDF data. In Blockeel, H., Kersting, K., Nijssen, S., Zelezný, F., eds.: ECML/PKDD (1). Volume 8188 of Lecture Notes in Computer Science., Springer (2013) 606–621
10. Kramer, S., Lavrac, N., Flach, P. In: Propositionalization Approaches to Relational Data Mining. Springer-Verlag (September 2001) 262–291
11. Nickel, M., Tresp, V., Kriegel, H.P.: A three-way model for collective learning on multi-relational data. In Getoor, L., Scheffer, T., eds.: ICML, Omnipress (2011) 809–816
12. LeCun, Y., Cortes, C.: The MNIST database of handwritten digits. http://yann.lecun.com/exdb/mnist/ (1998)

Probabilistic Latent-Factor Database Models

Denis Krompaß[1], Xueyian Jiang[1], Maximilian Nickel[2], and Volker Tresp[1,3]

[1] Ludwig Maximilian University of Munich, Germany
[2] Massachusetts Institute of Technology, Cambridge, MA
and Istituto Italiano di Tecnologia, Genova, Italy
[3] Siemens AG, Corporate Technology, Munich, Germany

Abstract. We describe a general framework for modelling probabilistic databases using factorization approaches. The framework includes tensor-based approaches which have been very successful in modelling triple-oriented databases and also includes recently developed neural network models. We consider the case that the target variable models the existence of a tuple, a continuous quantity associated with a tuple, multi-class variables or count variables. We discuss appropriate cost functions with different parameterizations and optimization approaches. We argue that, in general, some combination of models leads to best predictive results. We present experimental results on the modelling of existential variables and count variables.

1 Introduction

Tensor models have been shown to efficiently model triple-oriented databases [10] where the main goal is to predict the probability for the existence of a triple. Here we generalize the approach in several directions. First, we show that relations with any arity can be modeled, not just triple stores. Second, we show that any set of target variables that is associated with a triple can be modelled. As examples one might predict the rating of a user for an item, the amount of a specific medication for a patient, or the number of times that team A played against team B. In each of these cases a different likelihood model might be appropriate and we discuss different likelihood functions, their different parameterizations and learning algorithms. Third, we discuss a more general framework that includes recently developed neural network models [13, 1]. Finally, we argue that model combinations sometimes offer greater flexibility and predictive power. We present experimental results on the modelling of existential variables and count variables using different likelihood models.

The paper is organized as follows. In the next section we describe the probabilistic setting and in Section 3 we introduce the factorization framework and some specific models. In Section 4 we describe the learning rules and Section 5 contains our experimental results. Section 6 describes extensions. Section 7 contains our conclusions.

2 Probabilistic Database Models

2.1 Database Notation

Consider a database as a set of M relations $\{r^k\}_{k=1}^M$. A relation is a table with attributes as columns and tuples $E_i = \{e_{l(i,1)}, e_{l(i,2)}, \ldots, e_{l(i,L^k)}\}$ as rows where L^k is the number of attributes or the arity of the relation r^k. $l(i,i')$ is the index of the domain entity in tuple E_i in column i'. A relation r^k is closely related to the predicate $r^k(E_i)$, which is a function that maps a tuple to true (or 1) if the E_i belongs to the relation and to false (or 0) otherwise. We model a triple (s,p,o) as a binary relation p where the first column is the subject s and the second column is the object o.

2.2 Probabilistic Database Model

We now associate with each instantiated relation $r^k(E_i)$ a target quantity x_i^k. Formally we increase the arity of the relation by the dimension of x_i^k, so a binary relation would become a ternary relation, if x_i^k is a scalar. Here, the target x_i^k can model different quantities. It can stand for the fact that the tuple exists ($x_i^k = 1$) or does not exist ($x_i^k = 0$) i.e., we model the predicate. In another application E_i might represent a user/item pair and x_i^k is the rating of the user for the item. Alternatively, x_i^k might be a count, for example the number of times that the relation $r^k(E_i)$ has been observed. In the following we form predictive models for x_i^k; thus we can predict, e.g., the likelihood that a tuple is true, or the rating of a user for an item, or the number of times that relation $r^k(E_i)$ has been observed.

2.3 Likelihood Functions and Cost Functions

Convenient likelihood functions originate from the (overdispersed) exponential family of distributions.

Bernoulli. The Bernoulli model is appropriate if the goal is to predict the existence of a tuple, i.e., if we model the predicate. With $x_i^k \in \{0,1\}$, we model

$$P(x_i^k = 1 | \theta_i^k) = \theta_i^k$$

with $0 \leq \theta_i^k \leq 1$. From this equation we can derive the penalized log-likelihood cost function

$$\text{lossBe}_i^k = -(x_i^k + \alpha_i^k - 1)\log\theta_i^k - (\beta_i^k + K_i^k - x_i^k)\log(1-\theta_i^k). \tag{1}$$

Here, $K_i^k = 0$; $\alpha_i^k > 0$ and $\beta_i^k > 0$ are derived from the conjugate beta-distribution and can represent virtual data, in the sense that they represent $\alpha_i^k - 1$ additional observations of $x_i^k = 1$ and $\beta_i^k - 1$ additional observations of $x_i^k = 0$. The contribution of the prior drops out with $\alpha_i^k = 1, \beta_i^k = 1$.

Note that we have the constraints that $0 \leq \theta_i^k \leq 1$. A convenient re-parameterization can be achieved using the framework of the exponential family

of distributions which suggests the parametrization $\theta_i^k = \text{sig}(\eta_i^k)$, where the natural parameter η_i^k is unconstraint and where $\text{sig}(arg) = 1/(1+\exp(-arg))$ is the logistic function.

Gaussian. The Gaussian model can be used to predict continuous quantities, e.g., the amount of a given medication for a given patient. The Gaussian model is

$$P(x_i^k|\theta_i^k) \propto \exp -\frac{1}{2\sigma^2}(x_i^k - \theta_i^k)^2$$

where we assume that either σ^2 is known or is estimated as a global parameter in a separate process. With a Gaussian likelihood function we get

$$\text{lossG}_i^k = \frac{1}{2(\sigma_i^k)^2}(x_i^k - \theta_i^k)^2 + \frac{1}{2(\alpha_i^k)^2}(c_i^k - \theta_i^k)^2.$$

Note that the first term is simply the squared error. The second term is derived from the conjugate Gaussian distribution and implements another cost term, which can be used to model a prior bias toward a user-specified c_i^k. The contribution of the prior drops out with $\alpha_i^k \to \infty$.

Binomial. If the Bernoulli model represents the outcome of the tossing of one coin, the binomial model corresponds to the event of tossing a coin K times. We get

$$P(x_i^k|\theta_i^k) \propto (\theta_i^k)^{x_i^k}(1-\theta_i^k)^{K-x_i^k}.$$

The cost function is identical to the cost function in the Bernoulli model (Equation 1), only that $K_i^k = K - 1$ and $x_i^k \in \{0, 1, \ldots, K\}$ is the number of observed "heads".

Poisson. Typical relational count data which can be modelled by Poisson distributions are the number of messages sent between users in a given time frame. For the Poisson distribution, we get

$$P(x_i^k|\theta_i^k) \propto (\theta_i^k)^{x_i^k} \exp(-\theta_i^k) \tag{2}$$

and

$$\text{lossP}_i^k = -(x_i^k + \alpha_i^k - 1)\log\theta_i^k + (\beta_i^k + 1)\theta_i^k$$

with $x_i^k \in \mathbb{N}_0$; $\alpha_i^k > 0, \beta_i^k > 0$ are parameters in the conjugate gamma-distribution. The contribution of the prior drops out with $\alpha_i^k = 1, \beta_i^k = 0$. Here, the natural parameter is defined as $\theta_i^k = \exp(\eta_i^k)$. Note that the cost function of the Poisson model is, up to parameter-independent terms, identical to the KL-divergence cost function [3].

Multinomial. The multinomial distribution is often used for textual data where counts correspond to how often a term occurred in a given document. For the multinomial model we get

$$\text{lossM}_i^k = -(x_i^k + \alpha_i^k - 1)\log\theta_i^k$$

with $\theta_i^k \geq 0, \sum_i \theta_i^k = 1$. The natural parameter is defined as $\theta_i^k = \exp(\eta_i^k)$ and for observed counts, $x_i^k \in \mathbb{N}_0$. The contribution of the Dirichlet prior drops out with $\alpha_i^k = 1$.

Ranking Criterion. Finally we consider the ranking criterion which is used in the Bernoulli setting with $x_i^k \in \{0,1\}$. It is not derived from an exponential family model but has successfully been used in triple prediction, e.g., in [13]. Consider a binary relation where the first attribute is the subject and the second attribute is the object. For a known true tuple with $x_i^k = 1$ we define $\text{lossR}_i^k = \sum_{c=1}^{C} \max\left(0, 1 - \theta_i^k + \theta_{i,c}^k\right)$ where $\theta_{i,c}^k$ is randomly chosen from all triples with the same subject and predicate but with a different object with target 0. Thus one scores the correct triple higher than its corrupted one up to a margin of 1. The use of a ranking criterion in relational learning was pioneered by [12] as Bayesian Personalized Ranking (BPR) with a related ranking cost function of the form $\text{lossBPR}_i^k = \sum_{c=1}^{C} \log \text{sig}(\theta_i^k - \theta_{i,c}^k)$.

Interpretation. After modelling, the probability $P(x_i^k|\theta_i^k)$, resp. $P(x_i^k|\eta_i^k)$, can be interpreted as the plausibility of the observation given the model. For example, in the Bernoulli model we can evaluate how plausible an observed tuple is and we can predict which unobserved tuples would very likely be true under the model.

3 A Framework for Latent-Factor Models

3.1 The General Setting

We consider two models where all relations have the same arity L^k. In the *multi-task setting*, we assume the model

$$\theta^k_{E_i=\{e_{l(i,1)},e_{l(i,2)},\ldots,e_{l(i,L^k)}\}} = f_{w^k}\left(a_{l(i,1)}, a_{l(i,2)}, \ldots, a_{l(i,L^k)}\right). \tag{3}$$

Here a_l is a vector of $\gamma \in \mathbb{N}$ latent factors associated with e_l to be optimized during the training phase.[4] $l(i,i')$ maps attribute i' of tuple E_i to the index of the entity. This is a multi-task setting in the sense that for each relation r^k a separate function with parameters w^k is modelled.

In the *single-task setting*, we assume the model

$$\theta^k_{E_i=\{e_{l(i,1)},e_{l(i,2)},\ldots,e_{l(i,L^k)}\}} = f_{w}\left(a_{l(i,1)}, a_{l(i,2)}, \ldots, a_{l(i,L^k)}, \tilde{a}_k\right).$$

Note that here we consider a single function with parameter vector w where a relation is represented by its latent factor $\tilde{a}_k$.

In case that we work with natural parameters, we would replace $\theta^k_{E_i}$ with $\eta^k_{E_i}$ in the last two equations.

3.2 Predictive Models

We now discuss models for $f_{w^k}(\cdot)$ and $f_w(\cdot)$. Note that not only the model weights are uncertain but also the latent factors of the entities. We first describe

[4] Here we assume that the rank γ is the same for all entities; this assumption can be relaxed in some models.

tensor approaches for the multi-task setting and the single-task setting and then describe two neural network models.

Tensor Models for the Multi-task Setting. Here, the model is

$$f_{w^k}\left(a_{l(i,1)}, a_{l(i,2)}, \ldots, a_{l(i,L^k)}\right) \tag{4}$$

$$= \sum_{s_1=1}^{\gamma} \sum_{s_2=1}^{\gamma} \cdots \sum_{s_{L^k}=1}^{\gamma} w^k_{s_1,s_2,\ldots s_{L^k}} \left(a_{l(i,1),s_1} a_{l(i,2),s_2} \ldots a_{l(i,L^k),s_{L^k}}\right).$$

This equation describes a RESCAL model which is a special case of a Tucker tensor model with the constraint that an entity has a unique latent representation, independent of where it appears in a relation [10]. This property is important to achieve relational collective learning [10].

In the original RESCAL model, one considers binary relations with $L^k = 2$ (RESCAL2). Here A with $(A)_{l,s} = a_{l,s}$ is the matrix of all latent representations of all entities. Then Equation 4 can be written in tensor form as

$$\mathcal{F} = \mathcal{R} \times_1 A \times_2 A$$

with tensor $(\mathcal{F})_{l_1,l_2,k} = f_{w^k}(a_{l_1}, a_{l_2})$ and core tensor $(\mathcal{R})_{s_1,s_2,k} = w^k_{s_1,s_2}$.

Tensor Models for the Single-task Setting. Here, the model is

$$f_w\left(a_{l(i,1)}, a_{l(i,2)}, \ldots, a_{l(i,L^k)}, \tilde{a}_k\right) \tag{5}$$

$$= \sum_{s_1=1}^{\gamma} \sum_{s_2=1}^{\gamma} \cdots \sum_{s_{L^k}=1}^{\gamma} \sum_{t=1}^{\gamma} w_{s_1,s_2,\ldots s_{L^k},t} \left(a_{l(i,1),s_1} a_{l(i,2),s_2} \ldots a_{l(i,L^k),s_{L^k}} \tilde{a}_{k,t}\right).$$

Note that the main difference is that now the relation is represented by its own latent factor $\tilde{a}_k$. Again, this equation describes a RESCAL model. For binary relations one speaks of a RESCAL3 model and Equation 5 becomes

$$\mathcal{F} = \mathcal{R} \times_1 A \times_2 A \times_3 \tilde{A}$$

where $(\tilde{A})_{k,t} = \tilde{a}_{k,t}$ and the core tensor is $(\mathcal{R})_{s_1,s_2,t} = w_{s_1,s_2,t}$.

If $\tilde{a}_{k,t}$ is a unit vector with the 1 at $k = t$, then we recover the multi-task setting. If all weights are 0, except for "diagonal" weights with $s_1 = s_2 = \ldots = s_{L^k} = t$, this is a PARAFAC model and only a single sum remains. The PARAFAC model is used in the factorization machines [12]. In factorization machines, attributes with ordinal or real values are modelled by $\bar{a}_{z(i)} = z(i)\bar{a}$ where $z(i)$ is the value of the attribute in E_i and $\bar{a}$ is a latent factor vector for the attribute independent of the particular value $z(i)$.

Please note that the L^k-order polynomials also contain all lower-order polynomials, if we set, e.g., $a_{l,1} = 1$, $\forall l$. In the factorization machine, the order of the polynomials is typically limited to 1 or 2, i.e. all higher-order polynomials obtain a weight of 0.

Neural Tensor Networks. Here, the model is

$$f_{w^k,v^k}\left(a_{l(i,1)}, a_{l(i,2)}, \ldots, a_{l(i,L^k)}\right)$$
$$= \sum_{h=1}^{H} w_h^k \operatorname{sig}\left(\sum_{s_1=1}^{\gamma}\sum_{s_2=1}^{\gamma}\cdots\sum_{s_{L^k}=1}^{\gamma} v_{s_1,s_2,\ldots s_{L^k}}^{h,k}\left(a_{l(i,1),s_1} a_{l(i,2),s_2}\ldots a_{l(i,L^k),s_{L^k}}\right)\right).$$

The output is a weighted combination of the logistic function applied to H different tensor models. This is the model used in [13], where the $\tanh(\cdot)$ was used instead of the logistic function.

Google Vault Model. Here a neural network is used of the form

$$f_w\left(a_{l(i,1)}, a_{l(i,2)}, \ldots, a_{l(i,L^k)}, \tilde{a}_k\right)$$
$$= \sum_{h=1}^{H} w_h^k \operatorname{sig}\left(\sum_{s_1=1}^{\gamma} v_{1,s_1} a_{l(i,1),s_1} + \ldots + \sum_{s_{L^k}=1}^{\gamma} v_{L^k,s_{L^k}} a_{l(i,L^k),s_{L^k}} + \sum_{t=1}^{\gamma} \tilde{v}_t \tilde{a}_{k,t}\right).$$

The latent factors are simply the inputs to a neural network with one hidden layer. This model was used in [1] in context of the Google Knowledge Graph. It is related to tensor models for the single-task setting where the fixed polynomial basis functions are replaced by adaptable neural basis functions with logistic transfer functions.

4 Parameter Estimates

4.1 Missing Values

Complete Data. This means that for all relevant tuples the target variables are available.

Assumed Complete Data. This is mostly relevant when x_i^k is an existential variable, where one might assume that tuples that are not listed in the relation are false. Mathematically, we then obtain a complete data model and this is the setting in our experiments. Another interpretation would be that with sparse data $x_i^k = 0$ is a correct imputation for those tuples.

Missing at Random. This is relevant, e.g, when x_i^k represents a rating. Missing ratings might be missing at random and the corresponding tuples should be ignored in the cost function. Computationally, this can most efficiently be exploited by gradient-based optimization methods (see Section 4.3). Alternatively one can use α_i^k and β_i^k to implement prior knowledge about missing data.

Ranking Criterion. On the ranking criterion one does not really care if unobserved tuples are unknown or untrue, one only insists that the observed tuples should obtain a higher score by a margin than unobserved tuples.

4.2 Regularization

In all approaches the parameters and latent factors are regularized with penalty term $\lambda_A \|A\|_{\mathbf{F}}$ and $\lambda_W \|W\|_{\mathbf{F}}$ where $\|\cdot\|_{\mathbf{F}}$ indicates the Frobenius norm and where $\lambda_A \geq 0$ and $\lambda_W \geq 0$ are regularization parameters.

4.3 Optimizing the Cost Functions

Alternating Least Squares. The minimization of the Gaussian cost function lossG with complete data can be implemented via very efficient alternating least squares (ALS) iterative updates, effectively exploiting data sparsity in the (assumed) complete data setting [10, 7]. For example, RESCAL has been scaled up to work with several million entities and close to 100 relation types. The number of possible tuples that can be predicted is the square of the number of entities times the number of predicates: for example RESCAL has been applied to the Yago ontology with 10^{14} potential tuples [11].

Natural Parameters: Gradient-Based Optimization. When natural parameters are used, unconstrained gradient-based optimization routines like L-BFGS can be employed, see for example [6, 9].

Non-Negative Tensor Factorization. If we use the basis representation with θ_i^k parameters, we need to enforce that $\theta_i^k \geq 0$. One option is to employ non-negative tensor factorization which leads to non-negative factors and weights. For implementation details, consult [3].

Stochastic Gradient Descent (SGD). In principal, SGD could be applied to any setting with any cost function. In our experiments, SGD did not converge to any reasonable solutions in tolerable training time with cost functions from the exponential family of distributions and (assumed) complete data. SGD and batch SGD were successfully used with ranking cost functions in [12, 13] and we also achieved reasonable results with BPR.

5 Experiments

Due to space limitations we only report experiments using the binary multi-task model RESCAL2. We performed experiments on three commonly used benchmark data sets for relational learning:

Kinship 104 entities and $M = 26$ relations that consist of several kinship relations within the Alwayarra tribe.

Nations 14 entities and $M = 56$ relations that consist of relations between nations (treaties, immigration, etc). Additionally the data set contains attribute information for each entity.

UMLS 135 entities and $M = 49$ relations that consist of biomedical relationships between categorized concepts of the Unified Medical Language System (UMLS).

5.1 Experiments with Different Cost Functions and Representations

Here $x_i^k = 1$ stands for the existence of a tuple, otherwise $x_i^k = 0$. We evaluated the different methods using the area under the precision-recall curve (AUPRC) performing 10-fold cross-validation. Table 1 shows results for the three data sets ("nn" stands for non-negative and "nat" for the usage of natural parameters). In all cases, the RESCAL model with lossG ("RESCAL") gives excellent

Table 1. AUPRC for Different Data Sets

	Rand	RESCAL	nnPoiss	nnMulti	natBern	natPoiss	natMulti	SGD	stdev
Nations	0.212	0.843	0.710	0.704	**0.850**	**0.847**	0.659	0.825	0.05
Kinship	0.039	0.962	0.918	0.889	**0.980**	**0.981**	0.976	0.931	0.01
UMLS	0.008	**0.986**	0.968	0.916	**0.986**	0.967	0.922	0.971	0.01

Table 2. AUPRC for Count Data

	Rand	RESCAL	nnPoiss	nnMulti	natBin	natPoiss	natMulti	RES-P	stdev
Nations	0.181	0.627	0.616	0.609	**0.637**	0.632	0.515	**0.638**	0.01
Kinship	0.035	0.949	0.933	0.930	**0.950**	**0.952**	**0.951**	**0.951**	0.01
UMLS	0.007	0.795	0.790	0.759	**0.806**	**0.806**	0.773	**0.806**	0.01

performance and the Bernoulli likelihood with natural parameters ("natBern") performs even slightly better. The Poisson model with natural parameters also performs quite well. The performance of the multinomial models is significantly worse. We also looked at the sparsity of the solutions. As can be expected only the models employing non-negative factorization lead to sparse models. For the Kinship data set, only approximately 2% of the coefficients are nonzero, whereas models using natural parameters are dense. SGD with the BPR ranking criterion and AdaGrad batch optimization was slightly worse than RESCAL.

Another issue is the run-time performance. RESCAL with lossG is fastest since the ALS updates can efficiently exploit data sparsity, taking 1.9 seconds on Kinship on an average Laptop (Intel(R) Core(TM) i5-3320M with 2.60 GHz). It is well-known that the non-negative multiplicative updates are slower, having to consider the constraints, and take approximately 90 seconds on Kinship. Both the non-negative Poisson model and the non-negative multinomial model can exploit data sparsity. The exponential family approaches using natural parameters are slowest, since they have to construct estimates for all ground atoms in the (assumed) complete-data setting, taking approximately 300 seconds on Kinship. SGD converges in 108 seconds on Kinship.

5.2 Experiments on Count Data

Here $x_i^k \in \mathbb{N}_0$ is the number of observed counts. Table 2 shows results where we generated 10 database instances (worlds) from a trained Bernoulli model and generated count data from 9 database instances and used the tenth instance for testing. Although RESCAL still gives very good performance, best results are

Table 3. AUPRC for Combined Models

	RESCAL2	SUNS-S	SUNS-P	Combined	stdev
Nations	0.855	0.761	0.831	**0.886**	0.01
Kinship	0.960	0.916	0.004	**0.968**	0.01
UMLS	**0.979**	0.942	0.293	**0.980**	0.01

obtained by models more appropriate for count data, i.e., the binomial model and the Poisson model using natural parameters. The non-negative models are slightly worse than the models using natural parameters.

5.3 Probabilities with RESCAL

Due to its excellent performance and computational efficiency, it would be very desirable to use the RESCAL model with lossG and ALS, whenever possible. As discussed in [5], by applying post-transformations RESCAL predictions can be mapped to probabilities in Bernoulli experiments. For Poisson data we can assume a natural parameter model with $\alpha_i^k = 2$ and model $\tilde{x}_i^k = \log(1 + x_i^k)$ which leads to a sparse data representation that can efficiently be modelled with RESCAL and lossG. The results are shown as RES-P in Table 2 which are among the best results for the count data!

6 Extensions

6.1 SUNS Models

Consider a triple store. In addition to the models described in Section 3 we can also consider the following three model for $f(\mathit{subject}, \mathit{predicate}, \mathit{object})$

$$\sum_{m=1}^{\gamma}\sum_{u=1}^{\gamma} a_{\mathit{subject},m} a_u^{po}, \quad \sum_{m=1}^{\gamma}\sum_{u=1}^{\gamma} a_{\mathit{object},m} a_u^{sp}, \quad \sum_{m=1}^{\gamma}\sum_{u=1}^{\gamma} a_{\mathit{predicate},m} a_u^{so}.$$

These are three Tucker1 models and were used as SUNS models (SUNS-S, SUNS-O, SUNS-P) in [14]. a^{po}, a^{sp}, and a^{so} are latent representations of (p, o), (s, p), and (s, o), respectively.

6.2 Model Combinations

The different SUNS models and RESCAL models have different modelling capabilities and often a combination of several models gives best results [2, 8]. Table 3 shows the performance for the RESCAL model, two SUNS models and the performance of an additive model of all three models. For Nations, SUNS-P performs well and boosts the performance of the combined model. SUNS-P can model correlations between relations, e.g., between *likes* and *loves*.

7 Conclusions

We have presented a general framework for modelling probabilistic databases with factor models. When data are complete and sparse, the RESCAL model with a Gaussian likelihood function and ALS-updates is most efficient and highly scalable. We show that this model is also applicable for binary data and for

count data. Non-negative modelling approaches give very sparse factors but performance decreases slightly. An issue is the model rank γ. In [8] it has been shown that the rank can be reduced by using a combination of a factor model with a model for local interactions, modelling for example the triangle rule. Similarly, the exploitation of type-constraints can drastically reduce the number of plausible tuples and reduces computational load dramatically [4, 1].

Acknowledgements. M. N. acknowledges support by the Center for Brains, Minds and Machines, funded by NSF STC award CCF-1231216.

References

1. X. L. Dong, E. Gabrilovich, G. Heitz, W. Horn, N. Lao, K. Murphy, T. Strohmann, S. Sun, and W. Zhang. Knowledge vault: A web-scale approach to probabilistic knowledge fusion. In *KDD*, 2014.
2. X. Jiang, V. Tresp, Y. Huang, and M. Nickel. Link prediction in multi-relational graphs using additive models. In *SeRSy workshop, ISWC*, 2012.
3. D. Krompaß, M. Nickel, X. Jiang, and V. Tresp. Non-negative tensor factorization with rescal. In *Tensor Methods for Machine Learning, ECML workshop*, 2013.
4. D. Krompaß, M. Nickel, and V. Tresp. Factorizing large heterogeneous multi-relational-data. In *Int. Conf. on Data Science and Advanced Analytics*, 2014.
5. D. Krompaß, M. Nickel, and V. Tresp. Querying factorized probabilistic triple databases. In *ISWC*, 2014.
6. B. London, T. Rekatsinas, B. Huang, and L. Getoor. Multi-relational learning using weighted tensor decomposition with modular loss. In *arXiv:1303.1733*, 2013.
7. M. Nickel. *Tensor factorization for relational learning.* PhD-thesis, Ludwig-Maximilian-University of Munich, Aug. 2013.
8. M. Nickel, X. Jiang, and V. Tresp. Learning from latent and observable patterns in multi-relational data. In *NIPS*, 2014.
9. M. Nickel and V. Tresp. Logistic tensor factorization for multi-relational data. In *WSTRUC WS at the ICML*, 2013.
10. M. Nickel, V. Tresp, and H.-P. Kriegel. A three-way model for collective learning on multi-relational data. In *ICML*, 2011.
11. M. Nickel, V. Tresp, and H.-P. Kriegel. Factorizing yago: scalable machine learning for linked data. In *WWW*, 2012.
12. S. Rendle, L. B. Marinho, A. Nanopoulos, and L. Schmidt-Thieme. Learning optimal ranking with tensor factorization for tag recommendation. In *KDD*, 2009.
13. R. Socher, D. Chen, C. D. Manning, and A. Y. Ng. Reasoning with neural tensor networks for knowledge base completion. In *NIPS*, 2013.
14. V. Tresp, Y. Huang, M. Bundschus, and A. Rettinger. Materializing and querying learned knowledge. In *IRMLeS, ESWC workshop*, 2009.