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.

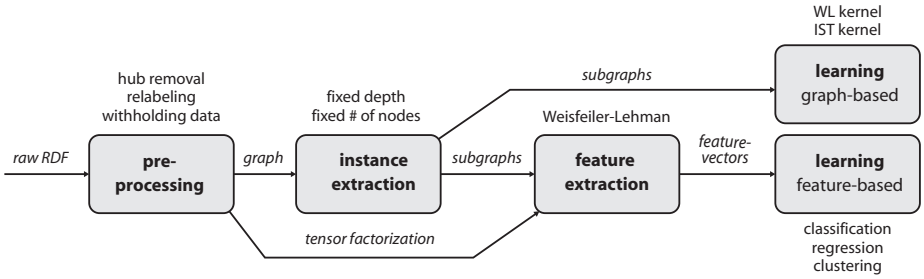

Fig. 1: An overview of a typical machine learning pipeline for RDF. Our aim here is not to provide a comprehensive framework, but to highlight some common steps.

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)