Predicting SPARQL Query Performance

Rakebul Hasan, Fabien Gandon

Predicting SPARQL Query Performance

Rakebul Hasan and Fabien Gandon

INRIA Sophia Antipolis, Wimmics, 2004 route des Lucioles - B.P. 93,
06902 Sophia-Antipolis Cedex, France,
`{hasan.rakebul,fabien.gandon}@inria.fr`

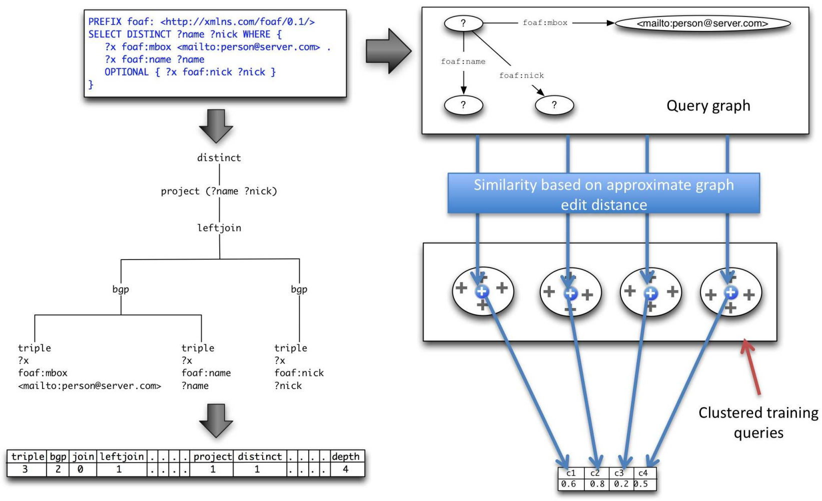
Abstract. We address the problem of predicting SPARQL query performance. We use machine learning techniques to learn SPARQL query performance from previously executed queries. We show how to model SPARQL queries as feature vectors, and use k-nearest neighbors regression and Support Vector Machine with the nu-SVR kernel to accurately (R^2 value of 0.98526) predict SPARQL query execution time.

1 Query Performance Prediction

The emerging dataspace of Linked Data presents tremendous potential for large-scale data integration over cross domain data to support a new generation of intelligent application. In this context, it increasingly important to develop efficient ways of querying Linked Data. Central to this problem is knowing how a query would behave prior to executing the query. Current generation of SPARQL query cost estimation approaches are based on data statistics and heuristics. Statistics-based approaches have two major drawbacks in the context of Linked Data [9]. First, the statistics (e.g histograms) about the data are often missing in the Linked Data scenario because they are expensive to generate and maintain. Second, due to the graph-based data model and schema-less nature of RDF data, what makes effective statistics for query cost estimation is unclear. Heuristics-based approaches generally do not require any knowledge of underlying data statistics. However, they are based on strong assumptions such as considering queries of certain structure less expensive than others. These assumptions may hold for some RDF datasets and may not hold for others. We take a rather pragmatic approach to SPARQL query cost estimation. We learn SPARQL query performance metrics from already executed queries. Recent work [1, 3, 4] in database research shows that database query performance metrics can be accurately predicted without any knowledge of data statistics by applying machine learning techniques on the query logs of already executed queries. Similarly, we apply machine learning techniques to learn SPARQL query performance metrics from already executed queries. We consider query execution time as the query performance metric in this paper.

2 Modeling SPARQL Query Execution

We predict SPARQL query performance metrics by applying machine learning techniques on previously executed queries. This approach does not require any statistics of the underlying RDF data, which makes it ideal for the Linked Data scenario. We use two types of query features: SPARQL algebra features and graph pattern features. We use frequencies and cardinalities of the SPARQL algebra operators [1], and depth of the algebra expression tree as SPARQL algebra features. Regarding graph patterns features, transforming graph patterns to vec-



```
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
SELECT DISTINCT ?name ?nick WHERE {
   ?x foaf:mbox <mailto:person@server.com> .
   ?x foaf:name ?name
   OPTIONAL { ?x foaf:nick ?nick }
}
```

triple	bgp	join	leftjoin	.	.	.	.	project	distinct	.	.	.	.	depth
3	2	0	1	.	.	.	.	1	1	.	.	.	.	4

c1	c2	c3	c4
0.6	0.8	0.2	0.5

distinct
project (?name ?nick)
leftjoin
bgp
bgp
triple ?x foaf:mbox <mailto:person@server.com>
triple ?x foaf:name ?name
triple ?x foaf:nick ?nick
?
foaf:mbox
<mailto:person@server.com>
foaf:name
foaf:nick
?
?
Query graph
Similarity based on approximate graph edit distance
Clustered training queries

Fig. 1. Example of extracting SPARQL feature vector from a SPARQL query.

tor space is not trivial because the space is infinite. To address this, we create a query pattern vector representation relative to the query patterns appearing in the training data. First, we cluster the structurally similar query patterns in the training data into K_{gp} number of clusters. The query pattern in the center of a cluster is the representative of query patterns in that cluster. Second, we represent a query pattern as a K_{gp} dimensional vector where the value of a dimension is the structural similarity between that query pattern and the corresponding cluster center query pattern. To compute the structural similarity between two query patterns, we first construct two graphs from the two query patterns, then compute the approximate graph edit distance – using a suboptimal algorithm [7] with $\mathrm{O}\left(n^3\right)$ computational complexity – between these two graphs. The structural similarity is the inverse of the approximate edit distance. We use the k-mediods [5] clustering algorithm to cluster the query patterns of training data. We use k-mediods because it chooses data points as cluster centers and allows using an arbitrary distance function. We use the same suboptimal graph edit distance algorithm as the distance function for k-mediods. Figure 1 shows an example of extracting SPARQL algebra features (left) and graph pattern features (right) from SPARQL query string.

[1] Algebra operators: http://www.w3.org/TR/sparql11-query/#sparqlAlgebra

3 Experiments and Results

We generate 1260 training, 420 validation, and 420 test queries from the 25 DBPSB benchmark query templates [6]. To generate queries, we assign randomly selected RDF terms from the DBpedia 3.5.1 dataset to the placeholders in the query templates. We run the queries on a Jena-TDB 1.0.0 triple store loaded with DBpedia 3.5.1 and record their query execution time. We exclude queries which do not return any result (queries from template 2, 16, and 21) and run more than 300 seconds (queries from template 20). We experiment with k-nearest neighbors (k-NN) regression [2] and Support Vector Machine (SVM) with the nu-SVR kernel for regression [8] to predict query execution time. We achieve an R^2 value of 0.9654 (Figure 2(a)) and a root mean squared error (RMSE) value of 401.7018 (Figure 2(b)) on the test dataset using k-NN (with $K_{gp} = 10$ and $k = 2$ selected by cross validation). We achieve an improved R^2 value of 0.98526 (Figure 2(c)) and a lower RMSE value of 262.1869 (Figure 2(d)) using SVM (with $K_{gp} = 25$ selected by cross validation). This shows that our approach can accurately predict SPARQL query execution time.

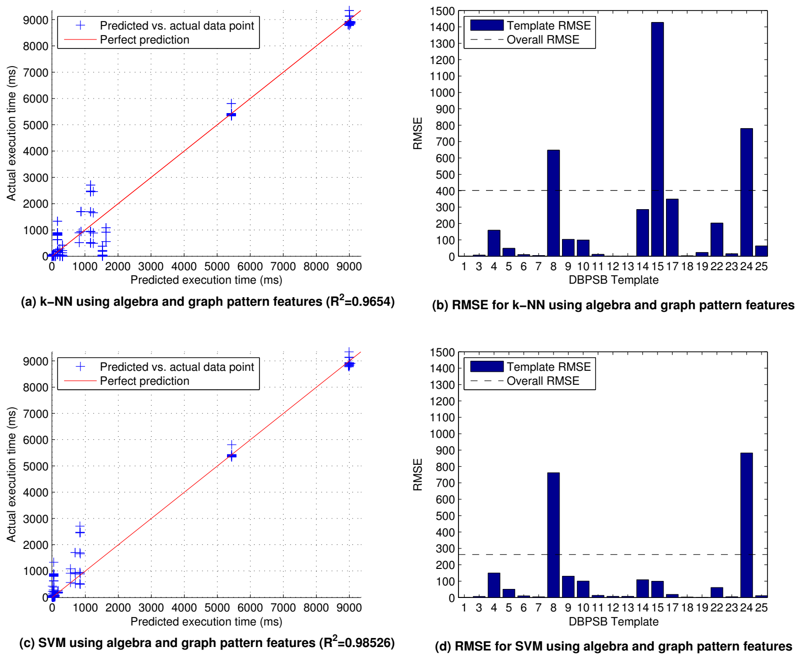


Fig. 2. Predictions for the test dataset with SPARQL algebra features and graph pattern features using k-NN ($K_{gp} = 10$ and $k = 2$) and SVM ($K_{gp} = 25$).

4 Conclusion and Future Work

We present an approach to predict SPARQL query execution time using machine learning techniques. We learn query execution times from already executed queries. This approach can be useful where statistics about the underlying data are unavailable We discuss how to model SPARQL queries as feature vectors, and show highly accurate results. In future, we would like to compare our approach to the existing SPARQL query cost estimation approaches in the context of Linked Data query processing.

Acknowledgments: This work is supported by the ANR CONTINT program under the Kolflow project (ANR-2010-CORD-021-02).

References

1. Akdere, M., Cetintemel, U., Riondato, M., Upfal, E., Zdonik, S.: Learning-based query performance modeling and prediction. In: Data Engineering (ICDE), 2012 IEEE 28th International Conference on. pp. 390–401 (2012)
2. Altman, N.: An introduction to kernel and nearest-neighbor nonparametric regression. The American Statistician 46(3), 175–185 (1992)
3. Ganapathi, A., Kuno, H., Dayal, U., Wiener, J.L., Fox, A., Jordan, M., Patterson, D.: Predicting multiple metrics for queries: Better decisions enabled by machine learning. In: Proceedings of the 2009 IEEE International Conference on Data Engineering. pp. 592–603. ICDE '09, IEEE Computer Society, Washington, DC, USA (2009)
4. Gupta, C., Mehta, A., Dayal, U.: PQR: Predicting query execution times for autonomous workload management. In: Proceedings of the 2008 International Conference on Autonomic Computing. pp. 13–22. ICAC '08, IEEE Computer Society, Washington, DC, USA (2008)
5. Kaufman, L., Rousseeuw, P.: Clustering by means of medoids. In: Dodge, Y. (ed.) Statistical Data Analysis based on the L1 Norm, p. 405416 (1987)
6. Morsey, M., Lehmann, J., Auer, S., Ngonga Ngomo, A.C.: Dbpedia SPARQL benchmark performance assessment with real queries on real data. In: Aroyo, L., et al. (eds.) The Semantic Web ISWC 2011, LNCS, vol. 7031, pp. 454–469. Springer Berlin Heidelberg (2011)
7. Riesen, K., Bunke, H.: Approximate graph edit distance computation by means of bipartite graph matching. Image Vision Comput. 27(7), 950–959 (Jun 2009)
8. Shevade, S.K., Keerthi, S.S., Bhattacharyya, C., Murthy, K.R.K.: Improvements to the SMO algorithm for SVM regression. Neural Networks, IEEE Transactions on 11(5), 1188–1193 (2000)
9. Tsialiamanis, P., Sidirourgos, L., Fundulaki, I., Christophides, V., Boncz, P.: Heuristics-based query optimisation for SPARQL. In: Proceedings of the 15th International Conference on Extending Database Technology. pp. 324–335. EDBT '12, ACM, New York, NY, USA (2012)