Query-based inferences in the Complex Data Management System

Luiz Gomes-Jr GOMESJR@IC.UNICAMP.BR
Institute of Computing – State University of Campinas (UNICAMP), Campinas – SP – Brazil

Rodrigo Jensen RODJEN@FCM.UNICAMP.BR
Faculty of Nursing – State University of Campinas (UNICAMP), Campinas – SP – Brazil

André Santanchè SANTANCHE@IC.UNICAMP.BR
Institute of Computing – State University of Campinas (UNICAMP), Campinas – SP – Brazil

Abstract

This paper describes how the Complex Data Management System (CDMS) enables query-based inferences over structure and unstructured data represented in its graph model. The CDMS offers querying and management mechanisms for data typical of complex networks. It enables flexible querying based on combinations of correlation metrics that capture properties of the topology of the underlying graph. This flexibility supports a range of information retrieval applications.

Here we show preliminary work on how the CDMS infrastructure can also be used in learning tasks. We envision a framework in which, for certain tasks, learning is indistinguishable from the conventional evolution of the database. Feature extraction and management is based on CDMS's *mapper* mechanism. Learned models are represented as queries, with combination of metrics and parameters fitted to the training data. We show preliminary experiments based on real data for a health diagnosis task.

1. Introduction

The Databases (DB) and Machine Learning (ML) fields have evolved to become essential parts of computational infrastructures. Although by means of different mechanisms, both fields answer queries or make inferences based on digital data representing real world entities and concepts. Their tasks are, however, typically undertaken as separated processes. Data from databases indirectly feed learning models whose inferred data is in turn stored in databases. Guiding the processes are experts in one area or the other, developing *ad hoc* solutions to connect the ends, and rarely overlapping technical skills.

On another level, both DB and ML fields have developed to handle increasingly complex data in terms of how data elements are correlated. Several ML tasks have to discover complex interactions among features. These tasks usually adopt graph-based models such as decision trees, neural networks, and, becoming popular more recently, hidden markov models (HMM), conditional random fields (CRF), relational markov networks (RMN), etc. Databases, likewise, have adopted higher complexity in their models, allowing for more flexible representation of relationships. This trend is evident in models that attempt to overcome limitations of the relational models, such as the object-oriented or semistructured models, and especially, the graph models that experienced renewed interest in the last decade.

In this paper, we focus on demonstrating the benefits of a tighter integration among the DB and ML fields. The two main points of our proposal are (i) making feature extraction part of the data lifecycle, i.e. materializing features as data elements, and (ii) mapping the supervised learning of classification models into a query composition problem. To tackle these points, we employ a new database system being defined by the authors. The Complex Data Management System (CDMS) aims at enabling querying and management of data typical of *complex networks*. The system adopts a graph data model to represent data from structured or unstructured sources. The CDMS also offers mechanisms to simplify the management of relationship lifecycle and, most importantly, a flexi-

ICML workshop on Structured Learning: Inferring Graphs from Structured and Unstructured Inputs (SLG 2013).

ble query model that can represent complex interactions among data elements. The query model allows parametrized definition of ranking metrics that are assessed based on properties of the topology of the underlying graph. This enables a level of expressiveness that can blur the line between querying and data analysis.

Our experiments demonstrate preliminary results based on a nursing diagnosis task over real data. We show how the classification task can be expressed as a declarative query in our query model. We also show how (manual) parameter tuning improves accuracy.

The main advantages of our approach are: (i) features become part of the database, evolving together with data, and (ii) the learned model is a declarative query, which gives insight into the important relationships among data and gives the opportunity for developers to tweak query parameters and scope with no need of retraining.

This paper is organized as follows: Section 2 describes related work in ML and complex networks. Section 3 defines the Complex Data Management System and introduces ML-related issues. Section 4 describes how ML tasks can be represented in our framework and presents experiments on real data. Section 5 concludes the paper.

2. Related Work

Relationship analysis and topological properties are central to both machine learning tasks and complex network analysis. If the outcome of a ML task depends on complex, non-independent interactions among features, it is important to employ adequate models that can capture the intricacies of the relationships. Decision trees, neural networks, HMMs, and CRFs (Lafferty et al., 2001) are models that aim at capturing these non-trivial associations in graphs representing features and outcomes. The more complex the graph model, the better its ability to capture non-linearities, and the bigger its computational costs.

More recent models have enabled collective reasoning, capturing associations among multiple instances of data and among related learning tasks. Examples of such models are the Markov Logic Networks (Richardson & Domingos, 2006) and Relational Markov Networks (Bunescu & Mooney, 2004). In this proposal, we also aim at capturing the correlations among elements and outcomes represented in a graph (the database itself). The most important distinction is that we want to represent the relevant correlations (i.e. the learned model) as a declarative query language.

The database framework used in our proposal is being developed to tackle issues associated with Complex Networks. In a complex network (Costa et al., 2007), the patterns defined by the interconnections are non-trivial, deviating substantially from cases where connections have the same probability (e.g. lattices or random graphs). The techniques developed for complex network analysis have become important resources in diverse applications such areas as systems biology, neuroscience, communications, transportation, power grids, and economics (Costa et al., 2011).

Our CDMS is aimed at enabling querying and management of what we define as *complex data*. Complex data is characterized when relationships are central to data analysis. In these cases, the graph formed by data entities (nodes) and relationships (links) present properties typical of complex networks. Our query language employs metrics that express correlations among data elements. These metrics are related to easily interpretable concepts such as relevance and reputation. Our hypothesis is that our language can, for some tasks, capture the same type of underlying patterns captured by graph-based learning models. Our data management mechanisms (e.g. *mappers*) can also be employed to simplify feature extraction and management.

3. Complex Data Management

The CDMS is aimed at providing adequate support for handling and querying complex data. It differs from typical DBMSs in four main aspects: (i) data model, (ii) query language, (iii) query evaluation, and (iv) data management mechanisms. Each of these items is described below, together with pertinent considerations related to ML.

3.1. Data model and data integration

The data in target CDMS applications typically do not comply to pre-defined schemas. The high number and diversity of the relationships require a model where relationships are first-class citizens. Graph models are obvious choices in these settings. Their flexible modeling characteristics enable easy mapping of most types of data. Nodes with immediate access to neighbors is also an important feature for the type of computation involved. The CDMS framework adopts weighted edge-labeled property multigraphs to encode complex data.

This graph data model is a convenient means to represent data from structured and unstructured sources.

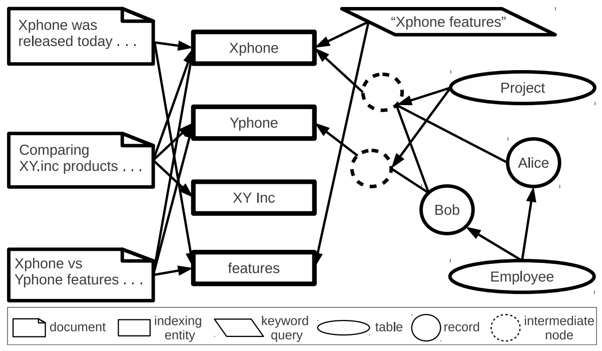

Figure 1. Data elements represented as a unified graph

```
SELECT ?doc                                                          a
 WHERE { ?doc :type :News }
RANK BY
 2 RELEVANCE OF ?doc TO TokenMapper("XPhone features")
 1 RELEVANCE OF ?doc TO :ProjectAlpha

START diag=node(*)                                                   b
   WHERE has(diag.Type) and diag.Type = "Diagnose"
   RETURN diag
RANK BY
   %r RELEVANCE OF diag TO node( %p ) WEIGHTED,
   %c CONNECTIVITY OF diag TO node( %p ) WEIGHTED
```

Figure 2. CDMS query examples (SPARQL and Cypher)

Mapping structured data to a graph is a straightforward process. Unstructured data, such as text documents, is a more challenging task. In our framework, it is convenient to represent individual unstructured elements as single nodes. The graph is then extended by extracting features from the added elements, which are themselves represented as nodes. Nodes representing the unstructured sources are linked to their extracted features through relationships created by *mappers*.

Figure 1 shows a simplified example to illustrate diverse elements represented as a unified graph. News articles about products are mapped into entities according to mappers that implement an indexing/annotation technique (e.g. topic modeling, named entity recognition, etc). A keyword query is likewise mapped into these entities, using the same mapper in query time. Relational data from tables (Project, Employee) are also mapped into nodes in the graph and also connected to the entities. More details on the use of mappers are presented in Section 3.4.

3.2. Query language

Our CDMS query language is intended to be flexible enough to allow correlation of data when little is known about how they are linked and organized. We developed a declarative query language that extends existing graph languages by introducing ranking based on a set of flexible correlation metrics. The ranking metrics proposed are: relevance, connectivity, reputation, influence, similarity, and context.

Our extension language is aimed at graph languages such as SPARQL[1] and Cypher[2]. We integrate the ranking metrics into existing query languages by adding a "RANK BY" clause. The clause enables an arbitrary combination of metrics that expresses the global ranking condition defined by the user.

[1]http://www.w3.org/TR/sparql11-query
[2]http://docs.neo4j.org

Figure 2a shows a typical query (in extended SPARQL) for data such as in Figure 1. The query retrieves documents relevant to the keyword query based on a simple tokenizer mapper (TokenMapper). Alternatively, the mapper could use more sophisticated strategies, such as topic modeling or entity recognition. The query also favors documents that are relevant to the project represented by URI :ProjectAlpha (prefixes are omitted). Figure 2b shows the query template (in Cypher) used in the classification task described in Section 4.

The raking metrics can specify several modifiers that restrict the subgraph used to assess the correlations. Among these modifiers are DEPTH, which limits the radius of the subgraph, and FOLLOW, which determines valid relationship labels to traverse. The tuning of these modifiers is directly related to the accuracy and computational efficiency of the queries.

The combination of the correlation metrics in a declarative querying setting enables a high level of flexibility and expressiveness for the applications to explore. We have employed our query language in information retrieval and recommendation tasks. Our goal here is to show how this query expressiveness can also be harnessed in order to represent some ML tasks. More details about the language, its design principles, and other applications can be found in (Gomes-Jr et al., 2013).

3.3. Processing model

Our abstractions for query evaluation fully support the query language while allowing for under-the-hood optimizations. We adopt a variation of the spreading activation (SA) model as our main abstraction for query evaluation. The model allows the specification of the ranking metrics that are the basis of our query language. The SA mechanism is based on traversing the network from a initial set of nodes, activating new nodes until certain stop conditions are reached. By controlling several aspects related to this activation flow, it is possible to infer and quantify the relationships of the initial nodes to the reached ones.

Our SA model is represented as $SA(N)$, defined by the parameters G, N, I, O, a, t, d, c, and l. A SA process starts with the N nodes initially activated with potential a. Output potentials for each node are calculated by the function $O(n) = I(n) * d$ (in the default case). The output potential is spread through all edges whose labels are in l. The potential for the reached nodes is calculated by function $I(n) = \sum_{i \in in(n)} O(i)$. For the next iteration, the potential is spread, restarting the process, as long as the current potential for reached nodes is higher than t and the radius of the activated network is lower than c. $v(n)$ represents the final value for the potential of n at the end of the process. A detailed description of the SA process can be found in (Gomes-Jr et al., 2013). We show below how some of the metrics are represented as SA processes.

Def. 1. $relevance(n, m) = v(SA(\{n\})_m)$,
with $O(n) = \dfrac{I(n) * d}{|out(n)|}$

Def. 2. $connectivity(n, m) = v(SA(\{n\})_m)$

Def. 3. $influence(n) = |SA(\{n\})|$

Def. 4. $context(m, n) = \dfrac{|SA(\{n\}) \cap SA(\{m\})|}{|SA(\{n\}) \cup SA(\{m\})|}$

These metrics capture topological properties of the underlying relationships that are associated with cognitive processes, making them easily understandable by users. **Relevance** between two nodes is a measure that encompasses correlation and specificity. Correlation is proportional to the number of paths linking the two nodes and inversely proportional to the length of the paths. Specificity favors paths with less ramifications. It is easy to observe that traditional tf*idf weighting over data as in Figure 1 is an instance of this definition (for trivial paths of length two). **Connectivity** between two nodes is a measure that assesses how interconnected two nodes are. The score is proportional to the number of paths linking the nodes in the network activated by the SA algorithm. **Influence** is a specialization of **reputation** where the only concern is the number of nodes reached from the origin. The topology of the graph – in/outdegree or cycles – do not influence the metric. **Context** and **similarity** measure the ratio of common elements surrounding two nodes. Context is a specialization of similarity where edge labels do not matter. The definitions and usage of other metrics can be found in (Gomes-Jr et al., 2013).

3.4. Relationship Management

Relationship creation is an important and defining operation for the described application scenarios. For example, several text indexing tasks, such as topic modeling, derive relationships between the text and more general concepts. In machine learning applications, elements are associated with features or classification categories, for example. In our framework, the creation of relationships is encapsulated in mappers. Mappers are very similar to *stored procedures*. What sets them apart are (i) their integrated use in our ranking queries, and (ii) how they are hooked in the databases's API so that any new data that matches the mapping criterion is passed through appropriate mappers.

Considering data as in Figure 1, a mapper is invoked whenever a document is added to the database. The mapper extracts features from the document – in this case, tokens or named entities. The features are materialized in the graph, with the new relationships marked as derived from their mapper. In query time (Figure 2a), the same mapper (TokenMapper) is invoked to extract features from the keyword query. The features and the keyword query are included as temporary nodes and relationships in the graph. Since all elements are represented in the same model, the query language can reference them as target for the correlation inferences.

4. CDMS for ML tasks

Our hypothesis is that the expressiveness of our query language coupled with the mapper mechanism can be an interesting way to represent some ML tasks. In our vision, learning could become indistinguishable from the natural process of database refinement and evolution. More data means better accuracy, i.e. more opportunities for exploring non-trivial patterns and for collective reasoning. Furthermore, refinements to the data, e.g. a classification mistake corrected by a database user, are immediately reflected in the data to be used by inference queries. Once all elements are represented in the same database model, inferences can be specified as queries in our query language. Optimizations in this scenario become a combination of sampling methods from ML and query optimization plans from DBs.

In our setting, mappers are used to extract features, which by default become part of the database. In classification tasks, the mappers also accumulate information about the correlation between features and outcomes (e.g. posterior probabilities for a bayesian mapper).

4.1. Experiments

We tested our approach over real diagnose data compiled for a previously unrelated project with the faculty of nursing at our university (Jensen et al., 2012). The data consists of matrices SD (for Symptoms $\times$ Diagnoses), PS (for Patients $\times$ Symptoms) and PD (for Patients $\times$ Diagnoses). SD correlates 62 symptoms to 28 diagnoses, all defined in a domain-specific protocol. The strength of the correlations $sd_{i,j} \in \{0, 0.25, 0.5, 0.75, 1\}$ were determined based on consensus in committees of experts. Similarly, PS correlates 6 patients (clinical cases) and their symptoms. The strength of the correlations $ps_{i,j} \in \{0, 0.25, 0.5, 0.75, 1\}$ were also determined by experts. Experts also provided most likely diagnoses for each patient, with strength $pd_{i,j} \in \{0, 0.25, 0.5, 0.75, 1\}$ (these were used as gold standards). The matrices were consolidated in a graph with the strength of the correlations becoming weights for the edges. It is important to notice that the weights defined by experts could be easily produced automatically by mappers if we had enough training data.

Figure 3 shows a sample of the graph with the weights of the associations represented as the thickness of the edges. As an example, the patient represented by the node 'Case 1' presents the symptom 'Fatigue' that was classified as having a 1.0 pertinence to the diagnosis 'Caregiver Role Strain' and 0.5 for 'Risk for Falls'. The final graph has 96 nodes and 324 relationships. Our goal was to suggest diagnoses based on the assessment of our metrics in the graph. The template for the queries issued for each patient is shown in Figure 2b. $\%r$ and $\%c$ are weights for the relevance and connectivity metrics, $\%p$ is the id of the node of the patient. To represent an instance of the query template we use the notation Rr:cC, meaning that the query was specified with weight r for the relevance metric and weight c for connectivity.

We analyzed the result rankings based on the diagnoses suggested by the committees of experts. We used the diagnoses with total correlation (1) to each patient. We run the queries to assess, for the sets of diagnoses for each patient, how many ranking places were needed to cover all diagnoses. The patients had 1 to 3 diagnoses matching our criteria. The connectivity metric required on average 1.5 extra ranking positions to cover the diagnosis, outperforming relevance which required 2. We then ran the query with different combinations of weights for the metrics, as shown in Figure 4. Combining the metrics improved the performance, eventually reducing the required extra positions to 1.17 on average.

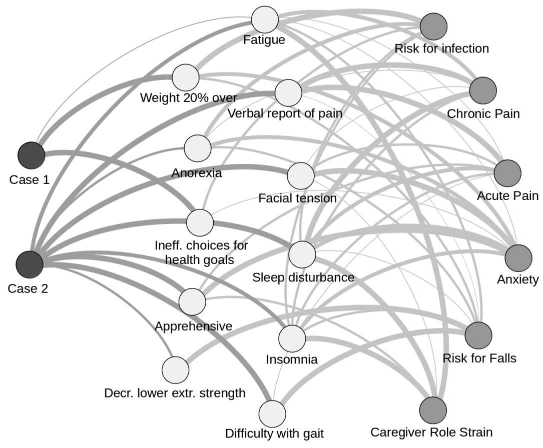

Figure 3. Sample subgraph of the dataset for the nursing diagnosis experiments

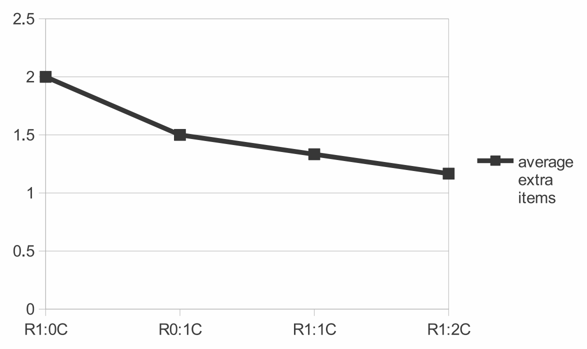

Figure 4. Average number of extra ranking items to cover the diagnosis from the expert

The analysis of these experiments suggest that characteristics from both metrics are important to produce good performing rankings, a trait we conjecture as being pervasive across tasks. The accuracy of our diagnosis suggestions is very practical, especially considering that we only analyzed a fraction of the information available for the experts that defined the gold standard – they also had access to patient's clinical report and a list of patient-specific risk factors.

Most importantly, we were able to produce the diagnosis recommendations from a very simple declarative query, in a development effort of a few minutes. The simple, manual query specification process described here is equivalent to fitting a classification function with the metrics and weights as parameters. In practice, our goal is to automatize the query specification process. The query, with ranking metrics tuned to maximize the likelihood of observing the training data, becomes the classification model.

5. Conclusion

There are many challenges and open questions related to this proposal. The search space for query composition is very large, including combinations for metric types, their weights, spreading activation parameters, and relationships to be traversed. Furthermore, query evaluation for large subgraphs and high frequency relationships is inefficient. We have, however, preliminary evidence that the performance challenges may be tractable. In our experiments (Gomes-Jr & Santanchè, 2013), we observed that good accuracy for the metrics can be achieve with low subgraph radii, and that relevant candidate values for SA parameters are in small ranges. Moreover, the declarative query setting offers opportunities for DB-like optimizations: we are proposing and testing a number of query optimization and approximation techniques. Using advanced sampling techniques will also contribute towards scalability.

Queries as materializations of learned models have several advantages compared to the abstract mathematical models learned in most ML tasks. The queries are easy for a user to interpret, tune and reuse. Moreover, the distribution of metrics, weights, and relationships provide insights on what are the most important patterns underlying the decision process. We also expect this approach to reduce, in many instances, the need for retraining as the database evolves.

We are just beginning to explore applications of our framework in learning tasks. We still need to assess the applicability of the language and the domain of tasks for which its expressiveness is adequate. Our model can only represent discrete features, but we believe it can be extended to include continuous features without a significant increase in complexity. This is an effort originated in the DB field that would greatly benefit from feedback and collaboration with ML researchers.

6. Acknowledgments

The authors would like to thank the Faculty of Nursing, UNICAMP. This work was partially financed by the Microsoft Research FAPESP Virtual Institute (NavScales project), CNPq (MuZOO Project and PRONEX-FAPESP), INCT in Web Science (CNPq 557.128/2009-9) and CAPES, with individual grants from CAPES and FAPESP (2010/10158-2).

References

Bunescu, R. C. and Mooney, R. J. Collective information extraction with relational markov networks. In *ACL*, pp. 438–445, 2004.

Costa, L., Oliveira Jr, O., Travieso, G., Rodrigues, F., Boas, P., Antiqueira, L., Viana, M., and Rocha, L. Analyzing and modeling real-world phenomena with complex networks: A survey of applications. *Advances in Physics*, 60:329–412, 2011.

Costa, L. D. F., Rodrigues, F. A., Travieso, G., and Boas, P. R. V. Characterization of complex networks: A survey of measurements. *Advances in Physics*, 56(1):167–242, 2007.

Gomes-Jr, L. and Santanchè, A. The Web Within: leveraging Web standards and graph analysis to enable application-level integration of institutional data. Technical Report IC-13-01, Institute of Computing, University of Campinas, January 2013.

Gomes-Jr, L., Jensen, R., and Santanchè, A. Towards query model integration: topology-aware, ir-inspired metrics for declarative graph querying. In *Second International Workshop on Querying Graph Structured Data*, 2013.

Jensen, R., Silveira, P. S. P., Ortega, N. R. S., and de M. Lopes, M. H. B. Software application that evaluates the diagnostic accuracy of nursing students. *Intl Journal of Nursing Knowledge*, 23:163–171, 2012.

Lafferty, J., McCallum, A., and Pereira, F. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proc. of 18th Int. Conf. on Machine Learning*, 2001.

Richardson, M. and Domingos, P. Markov logic networks. *Machine Learning*, 62(1-2):107–136, 2006.