Representation Independent Analytics Over Structured Data

Yodsawalai Chodpathumwan *ychodpa@illinois.edu* [*][1], Jose Picado *picadolj@oregonstate.edu* [*][2], Arash Termehchy *termehca@oregonstate.edu*[2], Alan Fern *afern@oregonstate.edu*[2] and Yizhou Sun *yzsun@ccs.neu.edu*[3]

[1]*School of EECS, Oregon State University, Corvallis, OR, USA*
[2]*College of Computer and Information Science, Northeastern University, Boston, MA, USA*
[3]*Department of Computer Science, University of Illinois, Urbana, IL, USA*

(Dated August 20, 2014)

Abstract Database analytics algorithms leverage quantifiable structural properties of the data to predict interesting concepts and relationships. The same information, however, can be represented using many different structures and the structural properties observed over particular representations do not necessarily hold for alternative structures. Thus, there is no guarantee that current database analytics algorithms will still provide the correct insights, no matter what structures are chosen to organize the database. Because these algorithms tend to be highly effective over some choices of structure, such as that of the databases used to validate them, but not so effective with others, database analytics has largely remained the province of experts who can find the desired forms for these algorithms. We argue that in order to make database analytics usable, we should use or develop algorithms that are effective over a wide range of choices of structural organizations. We introduce the notion of *representation independence*, study its fundamental properties for a wide range of data analytics algorithms, and empirically analyze the amount of representation independence of some popular database analytics algorithms. Our results indicate that most algorithms are not generally representation independent and find the characteristics of more representation independent heuristics under certain representational shifts.

1 Introduction

Over the last 20 years, users' information needs over structured data expanded from seeking exact answers to precise queries to finding entities or patterns *similar* to a given entity or pattern, discovering *interesting* entities and patterns, or predicting *novel* relations and concepts [13, 12, 9, 15, 24]. As part of its response, the research community proposed a multitude of supervised and unsupervised algorithms to solve exploration and analytics problems over structured data in different contexts, such as similarity query processing, inexact pattern matching, and concepts and relationship prediction [15, 24, 9, 12]. Since the properties of interesting and desirable answers are no longer precisely defined in the query, and in many cases there is no query at all in the traditional sense, these algorithms use intuitively appealing heuristics to choose, from among all possible answers, those that are most interesting, important, and likely to satisfy the user's information need.

The power of database exploration and analytics remains out of the reach of most users, however, as today's database exploration and analytics algorithms and tools are usable only by highly trained database analysts who can predict which algorithms are likely to be effective for particular representations of the data, or who are able to customize these algorithms to satisfy their information needs over a new database. To see why, consider the following examples.

Example 1.1 Relational Learning: *Given a database and training instances of a new target relation, relational learning algorithms attempt to induce general (approximate) definitions of the target in terms of existing relations. For example, given a database with student and professor information, the goal may be to induce a Datalog definition of a missing relation advisedBy(stud,prof) based on a training set of known student-advisor pairs. Since the space of possible definitions (e.g. all Datalog programs) is enormous, learning algorithms employ heuristics to search for effective definitions, which generally depend on the schema of the database. More generally, statistical relational learning algorithms[9] use the same heuristic mechanisms for structure learning, which in turn renders them schema dependent.*

As an example, Figure 1 shows the relations from three schemas for the UW-CSE[1] *database over students and professors, which is used as a common relational learning benchmark. The first and original schema was*

[*] These authors contributed equally to this work.
[1] http://alchemy.cs.washington.edu/data/uw-cse

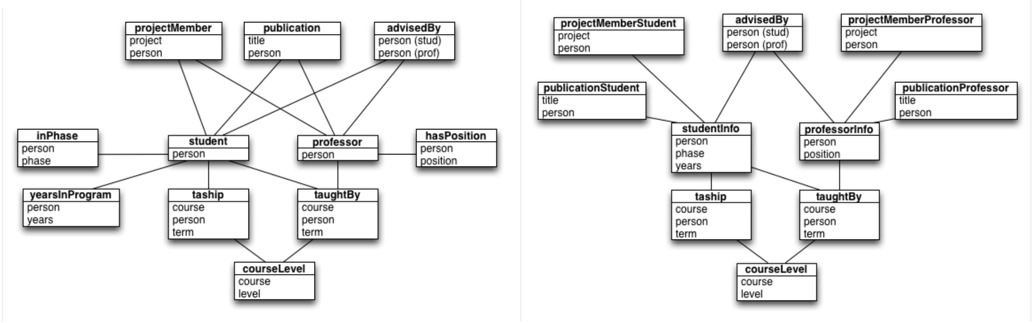
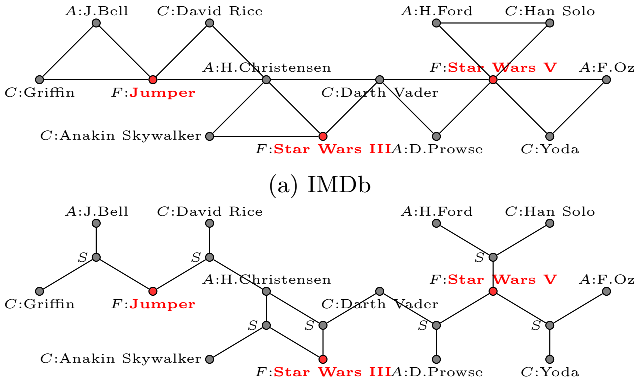


Figure 1: Various Schemas for the UW-CSE database.

designed by relational learning experts and is (almost) in sixth normal form, which is generally discouraged due to its poor usability and performance in query processing [1]. A database designer may use a schema closer to the alternative schemas in Figure 1. We used the popular learning algorithm FOIL [20] to induce a definition of advisedBy(stud,prof) for each of the two schemas, resulting in two very different definitions. The original schema yielded a more accurate definition based on co-authorship information, while the alternative schema led to a definition based on information about TA and teaching assignments.

Example 1.2 Graph Analytics: *Figure 2 (a) and (b) show excerpts of IMDb (imdb.com) and Freebase (freebase.com) about the same set of movies, their characters, and the actors who played them, respectively. IMDb and Freebase use different representations to express the same relationships between movies. Various link-based similarity search algorithms over graph databases, such as Random Walk with Restart (RWR) [24] and SimRank [15], use proximity-based heuristics. RWR evaluates how likely an entity will be visited if a random surfer starts and keeps re-starting from the query entity. SimRank evaluates the similarity between two entities according to how likely two random surfers will meet each other if they start from the two entities. RWR and SimRank find Star Wars III more similar to Star Wars V than to Jumper in Figure 2(a), but find Star Wars III to be more similar to Jumper in Figure 2(b).*



(a) IMDb

(b) Freebase

Figure 2: Fragments of IMDb and Freebase, where A, C, F, and S refer to *actor*, *character*, *film* and *starring*, respectively.

Generally, there is no canonical representation for a particular set of content and people often represent the same information in different structures [1, 8, 14]. For example, many researchers use DBLP (*dblp.uni-trier.de*), which is a publicly available bibliographical database about publications in computer science with rather a few types of entities, to evaluate their algorithms. We have observed that different researchers often represent this dataset differently. Fig. 3 shows two different representations for DBLP in two papers [22, 25] published in premier database venues.

Thus, users generally have to restructure their databases to some *proper representations*, in order to effectively use database analytics algorithms, i.e., deliver the insights that a domain expert would judge as relevant and important. To make matters worse, these algorithms do not normally offer any clear description of their desired representations and database analysts have to rely on their own expertise and/or do trial and error

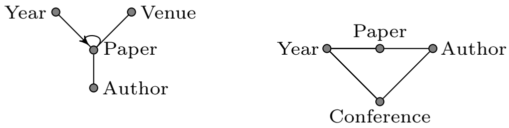


Figure 3: Different schemas for DBLP from [22] and [25].

to find such representations. Nevertheless, we want our database analytics algorithms to be used by ordinary users, not just experts. Further, the structure of large-scale databases constantly evolve [6], and we want to move away from the need for constant expert attention to keep exploration algorithms effective.

For similar reasons, current database analytics algorithms will not be well suited for Big Data, which is inherently heterogeneous and evolving as it obtains its content from many different data sources. Moreover, researchers often use analytics algorithms, such (statistical) relational learning and mining techniques, to solve various important data management problems, such as entity resolution, information extraction, and query processing [2, 9]. Thus, the issue of representation dependence appears in other areas of data management.

To cope with organizational heterogeneity and evolution in large-scale data, it is time to move beyond database analytics algorithms that are effective only over certain representations of the database. To this end, we propose a novel approach to database analytics that considers *representation independence*, i.e., the ability to deliver the same answers regardless of the choices of structure for organizing the data. **We argue for considering the robustness of an algorithm across various representations to better understand its usability.** Further, considering the authors' recent success in developing effective keyword search techniques that are robust across multiple representations of the same information [23], **we believe that it is possible to provide ordinary users with an arsenal of effective database analytics methods that are robust across multiple representations of the same information.** In particular:

- We introduce a novel framework that defines, quantifies, and analyzes the amount of representation independence of database analytics algorithms.
- We empirically study the representation independence of some popular relational learning and graph analytics algorithms. Our investigations indicates that most these algorithm largely depend on representational details, while a couple of them have reasonable amount of representation independence. It also offers promising insights for developing more representation independent algorithms.

This paper is organized as follows. Section 2 describes the related work. Section 3 introduces our proposal of representation independent data analytics. Section 4 contains our preliminary results on the amount of representation independent of some well known data analytics algorithms and Section 5 concludes the paper.

2 Related Work

Researchers have proposed declarative languages to achieve physical data independence for data analytics so developers do not worry about how the data is physically stored and what the execution environment is [13, 4, 10]. We, however, focus on the independence from logical representation.

The architects of relational model have argued for logical data independence, which oversimplifying a bit, means that an exact query should return the same answers no matter which logical schema is chosen for the data [1, 7]. The idea of representation independence extends the principle of logical data independence for database (DB) analytics algorithms. These ideas also differ in a couple of subtle but important issues. In our approach, the DB does not need to have a predefined schema, as many DB analytics algorithms explore DBs without any well-defined schema. One may achieve logical data independence by an affordable amount of experts' intervention, such as defining a set of views over the DB [1]. However, it generally takes more time and much deeper expertise to find the proper representation for a data analytics algorithm. Particularly, DB analytics applications normally contain more than a single algorithm [13], therefore, it will require a group of experts to achieve representation independence in this fashion. Hence, it is less likely to achieve representation independence via an affordable degree of expert's intervention.

Researchers have studied the issue of representational invariance in the context of probabilistic inference [21, 11]. They measure the invariance of a probabilistic inference method as its robustness against certain representational transformation(s). We, however, study this issue for supervised and unsupervised mining and learning algorithms. Researchers in this line of work measure the representation independence of a method under certain transformation as a binary value. In this paper, we propose non-binary metrics for representation independence, which make it easier to compare the robustness of various algorithms under same type of representational shifts. We also focus on representational variations and analytics algorithms over structured data models.

Researchers have analyzed the stability of some DB analytics algorithms against relatively small perturbations in the data [19, 9]. We also seek to instill robustness in DB analytics algorithms, but we are targeting robustness in a new dimension: robustness in the face of variations in the representation of data.

Finding a subset of relevant features from the data, is an important step in deploying learning algorithms [3]. When applied in this context, the idea of representation independence prefers features that are more robust against representational variations in the underlying DB.

We reported some initial results on the representation independence of similarity search algorithms over graphs in [5]. Current paper extends the idea of representation independence for general DB analytics algorithms and other data models, e.g. relational model, and provides the required formal framework for them. It compares other properties of algorithms, such as their running times, across various representations. It studies DB analytics algorithms for other problems, such as relational learning. We also analyze the amount of representation independence for graph analytics algorithms under some new representational shifts and DBs and find some algorithms to be representation independent under them, as opposed to the empirical studies in [5]. Finally, we analyze the characteristics of the representation independent algorithms for both graph analytics and relational learning.

3 Representation Independence

Intuitively, a representation independent algorithm should return the same answers across databases that represent the same information. We define the conditions under which different databases represent the same information. *Database transformations* have been used to model the relationship between different choices of structure in terms of the amount of information they can contain [8]. More formally, **transformation** τ over DB D is a (computable) function that maps D to another DB $\tau(D)$. If τ is *invertible*, $\tau(D)$ carries at least as much information as D. In other words, we can reconstruct the information available in D, given the transformed DB. For example, the transformation between IMDb and Freebase DBs in Figure 2 is invertible, as it replaces every *triangular* subgraph whose nodes represent entities of types *film*, *actor*, and *character* in a graph database with a *star* subgraph where these entities are connected via a single instance of type *starring.* If there is also an invertible transformation σ from $\tau(D)$ to D, D and $\tau(D)$ represent the same information and are called *equivalent.* We can similarly define the notion of information equivalence between two schemas. Let $I(S)$ be the set of all DBs with schema S. Given schemas S_1 and S_2, if there an *invertible* transformation τ: $I(S_1) \rightarrow I(S_2)$, and invertible transformation σ: $I(S_2) \rightarrow I(S_1)$, S_1 and S_2 are equivalent. If τ and σ belong to a family (set) of transformation T, e.g. vertical decompositions over relational schemas, we call schemas S_1 and S_2, and their mapped DBs, equivalent under T.

Many data analytics algorithms over structured data assume that their answers belong to a fixed *hypothesis language*, for example a relational learning algorithm may assume that its target concepts are Datalog expressions without recursion [9]. In order for an algorithm to deliver the same answers over equivalent DBs under a family of transformations T, T must also establish a bijective mapping between the hypothesis space over different representations. One may extend the work on query preservation [8], to check this property.

Some DB analytics algorithms, such as (statistical) relational learning and graph analytic methods, treat the values stored in the database as uninterpreted atomic object [9, 22, 24, 24]. Some invertible transformations may modify or perform some arithmetic operations on the values in the database. It is not reasonable to expect the algorithms that treat values as atomic objects to return the same answers over representational shifts that modify values. Hence, we focus on the algorithms and transformations that treat values as atomic objects and leave other types of transformations as future work.

Generally, algorithm A is *representation independent* under family of transformations T iff it returns the same answers over equivalent DBs under T for the same input. The answers across equivalent DBs may follow different representations, but convey the same information. The notion of input may be interpreted differently in different analytics problems For instance, it is a query for some DB analytics algorithms, such as finding similar entities, and a set of training data for supervised tasks, such as learning a novel relation.

Provided that algorithms A and B are representation independent under families of transformations T_A and T_B respectively, A is *more representation independent* than B if $T_A \supset T_B$. In other words, more representation independent algorithms are robust under a wider range of representational changes. For instance, a relational learning algorithm that is robust under both vertical and horizontal decompositions is more representation independent than the one that is robust only under vertical decomposition. Since perfect representation independence under a certain family of transformations may be hard to achieve for some analytic tasks, one may like to find the *most representation independent* algorithm for some analytic task under a certain set of transformations. Given a family of transformations T, algorithm A is *more representation independent* than algorithm B if A returns *more similar* answers than B across DBs that are equivalent under T. The amount of similarity between answers may be computed differently in different analytic tasks. For instance, the algorithms that find the most related entities to a given entity in a DB, return a ranked list of answers. Thus, one may use a distance metric for ranked lists to compute the similarities of the answers across equivalent DBs.

One may add additional constraints to the definition of representation independence in addition to the equality or similarity of answers. Our empirical studies of many DB analytics algorithms indicate that they need considerable amount of parameter (re-)tunning across various representations. Ideally, we would like a representation independent algorithm to have almost the same configuration across equivalent DBs, thus, a stronger type of representation independence is for an algorithm to have a minimal amount of reconfiguration across equivalent DBs. We have also found some algorithms to be very inefficient over certain representations, therefore, one may extend the definition of representation independence over family of transformations T such that the it returns the same or similar answers over equivalent DBs under T *within a reasonable amount of time.*

4 Empirical Study

One approach to create representation independence is to run an algorithm and over all representations of a validation subset of the DB and select the representation with the most accurate answers. Nonetheless, computing all representations of a DB is generally undecidable [8]. If confined within a particular family of transformations, DBs have normally an exponential number of representations, for example a relational table may have exponential number of distinct vertical decompositions. This number will be larger for the DBs that do not follow a fixed schema. As many algorithms need some time for parameter tunning under a new representation [16], it may take a prohibitively long time to find the best representation. Moreover, such a validation subset is not generally available for unsupervised methods. This approach also requires that the underlying DB be transformed to the desired representation, which may not be practical for a large and/or constantly evolving DB that is being used by varieties of algorithms, where each is effective over a different representation.

Another method is to define a *universal representation* to which all possible representations can be transformed and use/develop algorithms that are effective over this representation. The experience gained from the idea of universal relation, indicates that even for DBs with fixed schema such representation may not always exist [1]. Users also have to transform their DB to the universal representation, which may be quite complex considering the intricacies associated with defining such a representation and not practical for a large DB.

Hence, a more sustainable approach is to develop algorithms with reasonably high degree of representation independence. In this section, we analyze and compare the representation independence of popular relational learning and graph analytics algorithms to find characteristics of more representation independent heuristics.

4.1 Relational Learning

We evaluate the impact of representation on three popular relational learning algorithms: FOIL [20], Progol [17] and ProGolem [18]. FOIL is a greedy algorithm that induces one Datalog rule at a time by greedily specializing rules via adding literals to their bodies. Progol is similar to FOIL, but performs a more complete, non-greedy search over rule bodies within a restricted hypothesis space. ProGolem follows an alternative specific-to-general approach that searches for rules by combining pairs of overly specific rules into more general rules. We emulate both FOIL and Progol using Aleph[2], a well known Inductive Logic Programming (ILP) system. ProGolem is implemented in GILPS[3], another ILP system.

In Aleph, we use the following configuration: $noise$ = 100%, $minpos$ = 2, $search$ = $heuristic$, $evalfn$ = $compression$, $nodes$ = 10000, $clauselength$ = 5, and i = 2. We use the default values for the rest of the parameters except for *openlist* (beam), which we set to 1 to emulate FOIL and inf to emulate Progol. In ProGolem, we use the default configuration, except for the following parameters: $noise = 100\%$ and $i = 1$. The number of layers of new variables (parameter i) in Aleph is one unit higher than in ProGolem. Hence the values $i = 2$ in Aleph and $i = 1$ in ProGolem.

We run experiments using the UW-CSE and IMDb DBs. The UW-CSE data set consists of 12 relations, 2673 tuples, and 113 positive examples. We ignore the relation *tempAdvisedBy*. For both DBs, we generate negative examples using the closed-world assumption, and then sample these to obtain twice as many negative examples as positive examples. We divide the UW-CSE data set into 5 folds, each one corresponding to a different group of the CSE department, and we learn the relation *advisedBy(stud,prof)*, which indicates that student *stud* is advised by professor *prof.* We represent the DB using three equivalent schemas: the original schema, which is almost in 6th normal form, its transformed 4th normal form, and its denormalized schema,

[2] http://www.cs.ox.ac.uk/activities/machlearn/Aleph/aleph.html

[3] http://www.doc.ic.ac.uk/ jcs06/GILPS/

Algorithm	Metric	6NF	4NF	Denorm.
FOIL	Accuracy	**0.72**	**0.72**	0.64
	Precision	0.62	**0.64**	0.48
	Recall	0.70	**0.71**	0.41
	Time (s)	**2.44**	2.77	396.48
Progol	Accuracy	0.73	**0.77**	0.71
	Precision	**0.62**	**0.62**	0.59
	Recall	0.72	**0.81**	0.52
	Time (s)	51.06	**43.93**	226.39
ProGolem	Accuracy	**0.77**	0.74	0.75
	Precision	0.83	**0.86**	0.83
	Recall	**0.44**	0.32	0.36
	Time (s)	65.10	**56.31**	271.30

Table 1: Results of learning relations over UW-CSE.

shown in Figure 1. These schemas are mapped to each other via vertical decomposition (i.e. projection) and composition (i.e. join) transformations.

We download the relational DB for IMDb from JMDB[4], and obtain a subset of the DB to create 5 folds, where each fold contains information about 100 randomly selected movies. We remove the information about countries and genders from the DB, and learn the relations *countries(mov, ctry)*, which indicates that *mov* was filmed in *ctry*, and *femaleActor(actor)*, which indicates that *actor* is a female. We use two equivalent schemas for IMDb: its original schema, which is (almost) in 6NF, and an alternative schema that contains rather important attributes of a *movie* in a single table. We have picked these attributes based on the information in the main Web page of a movie in IMDb Website. One may prefer to use such a schema in order to generate and retrieve the main page of a movie fast. Since this schema is quite close to a 4th normal form, we denote it by 4NF in this section. These schemas are shown in Figure 4. Because it takes a long time to learn relations over more denormalized schemas of IMDb, we did not use them.

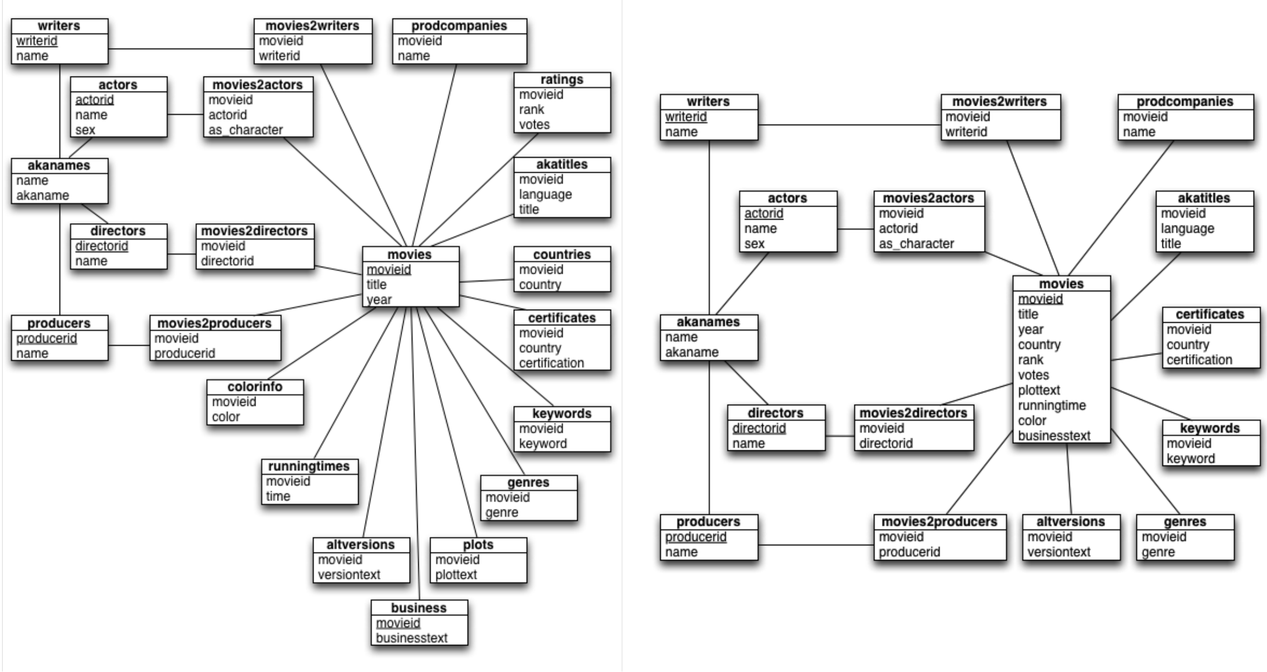


Figure 4: Schemas for the IMDb DB.

We evaluate accuracy, precision, recall, and running time, showing the average over 5-fold cross validation. Accuracy is the proportion of true results in all examples, precision is the proportion of true positives against all the positive results, and recall is the proportion of true positives against all positive examples. We have also measured the robustness of these methods in terms of the number of equivalent rules learned over different representations. We have found out that this metric may not reasonably reflect the amount of robustness for an algorithm as a slight variation in the rule definition will result in robustness of zero. Table 1 and 2 show the results for the UW-CSE and IMDb DBs, respectively.

[4] http://www.jmdb.de

Algorithm	Metric	6NF	4NF
FOIL	Accuracy	0.69	**0.71**
	Precision	0.48	**0.66**
	Recall	0.13	**0.20**
	Time (m)	129.10	**122.08**
Progol	Accuracy	0.70	**0.71**
	Precision	0.59	**0.65**
	Recall	0.17	**0.23**
	Time (m)	1269.39	**851.10**
ProGolem	Accuracy	**0.69**	**0.69**
	Precision	**0.87**	0.86
	Recall	0.08	**0.09**
	Time (m)	**16.22**	24.46

Table 2: Average results of learning relations over IMDb.

We first note that in general we see that ProGolem favors precision over recall, while the opposite is true for FOIL and Progol. This is expected since the specific-to-general search strategy of ProGolem typically leads to more specific rules than the alternative general-to-specific strategy of FOIL and Progol. This can also be observed by the fact that in general, FOIL and Progol learn more clauses (rules with more disjunctions) compared to ProGolem. This leads to an increase in recall for FOIL and Progol, and the opposite for ProGolem.

These results clearly show that the algorithms are not representation independent since the performance measures vary quite widely across different schemas. Further, there is not an ideal representation in general since the performance measures do not vary consistently with the level of normalization. A particularly notable observation is that the runtime of all learning algorithms is severely increased when using the denormalized UW-CSE schema. Such an increase in runtime would likely discourage users who use more denormalized schemas.

Another observation from the results is that the specific-to-general algorithm ProGolem appears to be more robust to representation change compared to the general-to-specific algorithms FOIL and Progol. For example, for UW-CSE the maximum difference in recall across representations is close to 0.3 for FOIL and Progol, but just 0.12 for ProGolem. Understanding the generality and reasons for this observation require further investigation. However, our current hypothesis is that specific-to-general algorithms are based on generalization operators that take bigger search steps and hence rely less on heuristic guidance compared to the specialization steps of FOIL and Progol.

Finally we can see that FOIL appears to be less robust than Progol. This is likely due to the fact that FOIL is the more greedy method and is hence more sensitive to its heuristics. As an example, under the denormalized schema of UW-CSE, FOIL learns the rule $advisedby(A, B) \leftarrow course(C, B, A, D, E)$, which indicates that the student is TA for a course offered by her advisor. This rule covers 12 positive examples, so FOIL greedily adds it to the hypothesis. However, even though Progol follows the same heuristic, it is able to explore a bigger space that allows it to find better rules. It is also worth noting that, because of practical reasons, we restrict the search space of FOIL and Progol to 10,000 nodes. Because Progol performs a more complete search, a less restricted search space may significantly increase its performance.

4.2 Graph Analytics

We have studied the representation independence of popular similarity search algorithms on IMDb and DBLP. For IMDb dataset, we create two DBs, small (s) and large (l), based on top 200 and top 5000 actors, their 1410 and 3592 associated movies produced in US from 2000 to 2012 with the rating of at least 6.0, and their 2864 and 22603 characters, respectively. We construct IMDb graph according to the schema in Figure 2(a). We then transform DB to the structure in Figure 2(b). For DBLP, we create two DBs, small (s) and large (l), based on top 200 and top 2000 authors from 15 and 50 conferences, and their 6106 and 17116 publications from 2005-2014, respectively. We construct the graph of DBLP based on the structure in Figure 3(a). We transform DBLP graph according to a schema of SIGMOD Record[5] where all authors of each paper is grouped under the node *authors*. We also transform DBLP graph according to the DBLP schema developed by L3S[6], which adds a *proceedings* node between associated *conference* and *year*, and a link between *paper* and *proceedings* if the paper appears in the proceedings. The size of each data graph and its transformation is shown in Table 3. We randomly select 50 actors and 50 authors as queries.

We compare similarity ranking between the original DBs and their transformations using RWR [24], SimRank [15], and PathSim [22]. Similar to RWR and SimRank, PathSim is another link-based similarity function, which captures the similarity between two entities according to the normalized path count following a specified meta-path between them, where a meta-path is a sequence of original relations that denote a new relation between the connected entities. Since the measurement of PathSim requires a meta-path to semantically indicate the

[5] cs.washington.edu/research/xmldatasets/

[6] dblp.l3s.de/d2r/

DB	Transformation	|V|	|E|
IMDb$_s$	-	4474	9369
	T_{Freebase}	7681	9621
IMDb$_l$	-	31195	176984
	T_{Freebase}	61482	181722
DBLP$_s$	-	8158	35213
	T_{SIGMOD}	14264	41290
	T_{L3S}	8290	41603
DBLP$_l$	-	19176	136216
	T_{SIGMOD}	36292	170354
	T_{L3S}	19511	171888

Table 3: Size of IMDb and DBLP data graphs

	T_{Freebase}		T_{SIGMOD}		T_{L3S}	
Algorithm	IMDb$_s$	IMDb$_l$	DBLP$_s$	DBLP$_l$	DBLP$_s$	DBLP$_l$
RWR	0.303	0.530	0.548	0.333	0.231	0.209
SimRank	0.363	0.498	0.533	0.410	0.195	0.215
PathSim	0.454	0.557	**0**	**0**	**0.060**	**0.022**
RWR	0.211	0.389	0.527	0.666	0.359	0.401
SimRank	0.435	0.426	0.526	0.716	0.389	0.386
PathSim	0.131	0.332	**0**	**0**	**0.023**	**0.027**

Table 4: Average ranking difference for top 10 (top table) and top 50 (bottom table) answers over IMDb and DBLP.

meaning of similarity between entities in a graph, we pick meta-paths that represent the same or similar relations in both original and transformed DBs. We use meta-path AFA in original IMDb graph, and $ASFSA$ in the transformed graph of IMDb. We use $APCPA$ in original DBLP graph, $AGPCPGA$ in the graph using SIGMOD schema, and $APRCRPA$ in the graph using L3S schema, where A, C, G, P and R refer to *author*, *conference*, *authors*, *paper* and *proceedings*, respectively.

We use normalized Kendall's tau metric [23] to measure the difference between two ranked lists, which is 0 if two lists are identical and 1 if one is the reverse of the other. Ranking methods usually return many answers but users focus mainly on top ranked results. Hence, we compare the top 10 or 50 answers for each query.

Table 4 shows the average ranking difference of top 10 and top 50 answers between the original and the transformed DBs of IMDb and DBLP. Overall, none of the methods are representation independent under all transformations. While keeping the same information, the transformations introduced in this experiment change the topological structures of the data graphs and directly affect the computation of both RWR and SimRank. Furthermore, since both SimRank and RWR use rather global information in the graph, the aggregated topological modifications caused by the transformations largely affects their rankings. On the other hand, the ranking difference of PathSim is comparable to RWR and SimRank for IMDb, but almost minimal for DBLP. This results shows that achieving representation independence under certain families of transformations is possible. Based on the robustness of PathSim on DBLP dataset, we believe that the underlying idea of PathSim is a promising direction to attain both representation independence and effectiveness, due to the following reasons. First, the computation of PathSim is local and constrained by a given meta-path. If transformation does not affect the part of the graph used in the computation, then the algorithm will be robust. Second, PathSim uses meta-paths to capture the semantic of relationships between entities. By using meta-paths that represent the same type of relationships in equivalent DBs, we can recover (almost) the same results over these DBs. If a given meta-path in one DB can be transformed to a meta-path in another graph, PathSim most likely provide the same rankings over equivalent DBs. Further, PathSim is shown to be generally more effective than other similarity algorithms [22].

However, semantically equivalent meta-path in equivalent DBs may not exists under certain representational shifts. For example, there is not any meta-path in the Freebase DB that has the exact semantic as meta-path AFA in IMDb. This is because in Freebase all paths that connect *actor* and *film* must pass through the node *starring*, S, which is unique per character. But there is only one path between each actor and each film in IMDb regardless of the number of characters the actor has played in the film. This causes the ranking differences for PathSim over IMDb DBs. Extending PathSim to resolve this problem is an interesting future work.

5 Conclusion and Future Work

Since current DB analytics algorithms generally rely on representational details of DBs, they are effective only over certain representations of the data. We argued that a usable algorithm should deliver the same results no matter which representation is chosen for the data and studied the fundamental properties of this notion. Our empirical studies provided new insights on the characteristics of some robust DB analytics algorithms. We plan to develop more representation independent algorithms for the discussed problems.

References

[1] S. Abiteboul, R. Hull, and V. Vianu. *Foundations of Databases.* Addison-Wesley, 1994.

[2] A. Abouzied, D. Angluin, J. Hellerstein, C. Papadimitriou, and A. Silberschatz. Learning and verifying quantified boolean queries by example. In *PODS*, 2013.

[3] M. Anderson, D. Antenucci, V. Bittorf, M. Burgess, M. Cafarella, A. Kumar, F. Niu, Y. Park, C. Re, and C. Zhang. Brainwash: A Data System for Feature Engineering. In *CIDR*, 2013.

[4] V. R. Borkar, Y. Bu, M. J. Carey, J. Rosen, N. Polyzotis, T. Condie, M. Weimer, and R. Ramakrishnan. Declarative Systems for Large-Scale Machine Learning. *IEEE Data Eng. Bull.*, 35(2), 2012.

[5] Y. Chopathumwan, A. Termehchy, Y. Sun, A. Aleyasin, and J. Picado. Toward representation independent similarity search. In *GRADES*, 2014.

[6] J. Cohen, B. Dolan, M. Dunlap, J. Hellerstein, and C. Welton. MAD Skills: New Analysis Practices for Big Data. *PVLDB*, 2(2), 2009.

[7] E. Codd. Does Your DBMS Run By the Rules? *ComputerWorld*, 1985.

[8] W. Fan and P. Bohannon. Information Preserving XML Schema Embedding. *TODS*, 33(1), 2008.

[9] L. Getoor and B. Taskar. *Introduction to Statistical Relational Learning.* MIT Press, 2007.

[10] A. Ghoting, R. Krishnamurthy, E. Pednault, B. Reinwald, V. Sindhwani, S. Tatikonda, Y. Tian, and S. Vaithyanathan. SystemML: Declarative Machine Learning on MapReduce. In *ICDE*, 2011.

[11] J. Y. Halpern and D. Koller. Representation Dependence in Probabilistic Inference. *JMLR*, 21, 2004.

[12] J. Han, M. Kamber, and J. Pei. *Data Mining: Concepts and Techniques.* Morgan Kaufmann, 2011.

[13] J. M. Hellerstein, C. Re, F. Schoppmann, Z. D. Wang, E. Fratkin, A. Gorajek, K. S. Ng, C. Welton, X. Feng, K. Li, and A. Kumar. The MADlib Analytics Library or MAD Skills, the SQL. *PVLDB*, 5(12), 2012.

[14] R. Hull. Relative Information Capacity of Simple Relational Database Schemata. *SICOMP*, 15(3), 1986.

[15] G. Jeh and J. Widom. SimRank: A Measure of Structural-context Similarity. In *KDD*, 2002.

[16] T. Kraska, A. Talwalkar, J. Duchi, R. Griffith, M. J. Franklin, , and M. Jordan. MLbase: A Distributed Machine-learning System. In *CIDR*, 2013.

[17] S. Muggleton. Inverse Entailment and Progol. *New Gen. Computing, Special issue on ILP*, 1995.

[18] S. Muggleton, J. Santos, and A. Tamaddoni-Nezhad. A System Based on Relative Minimal Generalisation. In *ILP*, 2009.

[19] A. Ng, A. Zheng, and M. Jordan. Stable Algorithms for Link Analysis. In *SIGIR*, 2001.

[20] J. R. Quinlan. Learning Logical Definitions From Relations. *Machine Learning*, 1990.

[21] W. C. Salmon. On Vindicating Induction. *Philosophy of Science*, 30(3), 1963.

[22] Y. Sun, J. Han, X. Yan, S. Yu, and T. Wu. PathSim: MetaPath-Based Top-K Similarity Search in Heterogeneous Information Networks. In *VLDB*, 2011.

[23] A. Termehchy, M. Winslett, Y. Chodpathumwan, and A. Gibbons. Design Independent Query Interfaces. *TKDE*, 24(10), 2012.

[24] H. Tong, C. Faloutsos, and J. Pan. Fast Random Walk with Restart and its Applications. In *ICDM*, 2006.

[25] P. Zhao, J. Han, and Y. Sun. P-Rank: A Comprehensive Structural Similarity Measure over Information Networks. In *CIKM*, 2009.