Provenance for Non-Experts

Daniel Deutch, Nave Frost, Amir Gilad
Tel Aviv University

Abstract

The flourish of data-intensive systems that are geared towards direct use by non-experts, such as Natural Language question answering systems and query-by-example frameworks calls for the incorporation of provenance management. Provenance is arguably even more important for such systems than for "classic" database application. This is due to the elevated level of uncertainty associated with the typical ambiguity of user specification (e.g. phrasing questions in Natural Language or through examples). Existing provenance solutions are not geared towards the non-experts, and the typical complexity and size of their instances render them ill-suited for this goal. We outline in this paper our ongoing research and preliminary results, addressing these challenges towards developing provenance solutions that serve to explain computation results to non-expert users.

1 Introduction

In the context of data-intensive systems, data provenance captures the way in which data is used, combined and manipulated by the system. Provenance information can for instance be used to reveal whether data was illegitimately used, to assess the trustworthiness of a computation result, or to explain the rationale underlying the computation. As data-intensive systems constantly grow in use, in complexity and in the size of data they manipulate, provenance tracking becomes of paramount importance. In its absence, it is next to impossible to follow the flow of data through the system. This in turn is extremely harmful for the quality of results, for enforcing policies, and for the public trust in the systems.

The focus of the present paper is on provenance tracking and presentation, geared towards the *non-expert* user. There is a large body of research and development on database interfaces for non-experts, such as Natural Language Interfaces [25, 26, 32, 39, 40] or exploratory systems such as query-by-example frameworks [1, 12, 29, 33, 36, 37, 42, 45]. Provenance is arguably even more important for such systems than it is for "classic" database applications, since uncertainty is far greater: there may be errors in the way the user has phrased the question and in the way the system has understood what she asked; even in the absence of errors, there are usually many valid queries that are consistent with the user's input. To this end, there is typically a phase of interaction, where the system refines its set of possible queries based on user feedback; provenance could potentially be highly useful for this phase as well as for the end result: it could allow users to understand what is the inferred query or set of candidate queries and whether they fit their intended semantics.

One may consider the use of existing provenance solutions for this purpose. After all, many of the above mentioned interfaces for non-experts eventually compile the user input into a formal query. Can we use a

Bulletin of the IEEE Computer Society Technical Committee on Data Engineering

provenance model designed for the underlying query language? The challenge is in that existing provenance solutions are not suitable for presentation to non-experts but rather focus on provenance storage through an internal representation and then its use for analysis. Specifically, the resulting provenance is typically both too complex and too large-scale to allow for its direct presentation to non-experts. This requires the development of dedicated solutions.

This paper outlines our recent and ongoing work on provenance tracking and presentation that are geared towards non-expert users. We discuss three application domains, as follows. The first two are database interfaces for non-experts of different flavours, namely Natural Language Interfaces and query-by-example frameworks. The third application domain is very different, involving explanations for results of Machine Learning models. We next identify several overarching principles that apply across domains and allow to reduce the complexity and scale of provenance, towards its presentation to non-experts.

- We claim that in order to be understandable to non-expert users, there should be a tight **coupling between the way provenance information is presented, and the standard user interaction with the system**. For example, if the user asks questions in Natural Language, then it is natural to present provenance information in a similar form; in a completely different domain, if one uses an image tagging application, then it is natural to depict provenance information as a layer on top of the image. This principle was proven successful in the context of (coarse-grained) workflow provenance [19, 38], where provenance is represented and shown in a graphic manner that is very similar to the way developers design the workflow itself; we claim that it is a key principle in the context of provenance presentation for non-experts as well.

- The sheer size of provenance calls for solutions that **summarize** it. Provenance summarization has been studied in multiple contexts, where summaries may be "lossless", i.e. involve no loss of information, as is the case with the work on factorized representation [4, 24, 31], or "lossy" as in [2, 13, 34]. In the context of non-experts, summarization needs not only to be concise but also to "make sense" to the non-expert. For instance, we demonstrate below that in the context of NL provenance, some of the possible summarizations may naturally be translated to semantically meaningful sentences, while others may not.

- Another approach for addressing provenance size is to track and present only portions of it, in a **selective** manner. Depending on the intended use, full provenance information may be unnecessary: for instance, if provenance is tracked with the goal of allowing users to distinguish between candidate formal queries generated by the system, then showing provenance for a "representative" sample of query outputs may suffice. Here again, the choice of samples may be based on the user interaction: for instance, we may show provenance for the examples that the user has given (and is thus familiar with), to demonstrate the logic captured by the inferred query; additional examples that show further diverse facets of the query may be selected as well.

In the rest of this paper we overview our results achieved so far in this area. In Section 2 we overview our solution from [13, 14] on Natural Language Provenance. In Section 3 we highlight the potential of incorporating provenance to query-by-example frameworks [15, 16]. In Section 4 we discuss explaining to non-experts the results of Machine Learning models, and outline an approach for achieving that. We conclude in Section 5.

2 Natural Language Explanations

Developing Natural Language (NL) interfaces for database systems has been the focus of multiple lines of research (see e.g. [3, 25, 26, 40]). Users who view results computed by such systems may also be interested in the explanations for these results, i.e. *why* does each answer qualify to the query criteria. Such explanations could greatly enrich the answers, as well as provide means for the user to understand what is the underlying formal query compiled by the system and whether it matches the user intention.

As an example, consider the Microsoft Academic Search database (`http://academic.research.microsoft.com`) and consider the NL query in Figure 1a. A state-of-the-art NL query engine, `NaLIR` [26], is able to transform this NL query into the SQL query also shown (as a Conjunctive Query) in Figure 1b. When evaluated using a standard database engine, the query returns the expected list of organizations. However, the answers (organizations) in the query result lack *justification*, which in this case would include the authors affiliated with each organization and details of the papers they have published (their titles, their publication venues and publication years). Such additional information can lead to a richer answer than simply providing the names of organizations: it allows users to also see relevant details of the qualifying organizations. Provenance information is also valuable for validation of answers: a user who sees an organization name as an answer is likely to have a harder time validating that this organization qualifies as an answer, compared to a setting where she is shown the full details of publications. Our approach is to present *provenance information for answers of NL queries, again as sentences in natural language*. Continuing our running example, Figure 1c shows one of the answers outputted by our system in response to the NL query in Figure 1a.

This solution [14] consists of the following key components.

"Return the organization of authors who published papers in database conferences after 2005"

(a) NL Query

```
query(oname) :-
org(oid, oname),
conf(cid, cname),
pub(wid, cid, ptitle, pyear),
author(aid, aname, oid),
domainConf(cid, did),
domain(did, dname),
writes(aid, wid),
dname = 'Databases',
pyear > 2005
```

(b) CQ Q

"Tel Aviv University is the organization of Tova Milo who published 'OASSIS...' in SIGMOD in 2014"

(c) Answer For a Single Assignment

Figure 1: NL Query, CQ Q, and an example NL Answer

Provenance Tracking Based on the NL Query Structure A first key idea in our solution is to leverage the *NL query structure* in constructing NL provenance. In particular, we modify `NaLIR` so that we store exactly which parts of the NL query translate to which parts of the formal query. Then, we evaluate the formal query using a provenance-aware engine (we use `SelP` [17]), further modified so that it stores which parts of the query "contribute" to which parts of the provenance. By composing these two mappings (text-to-query-parts and query-parts-to-provenance) we infer which parts of the NL query text are related to which provenance parts. Finally, we use the latter information in an "inverse" manner, to translate the provenance to NL text. We show the construction by example and refer the reader to [14] for further details.

Example 1: Re-consider our running example query and consider the database in Table 1. The assignments to the query are represented in Figure 2 as a DNF expression. Each of the 6 clauses stands for a different assign-

```
(oname,TAU)∧(aname,Tova M.)∧(ptitle,OASSIS...)∧(cname,SIGMOD)∧(pyear,14')∨
(oname,TAU)∧(aname,Tova M.)∧(ptitle,Querying...)∧(cname,VLDB)∧(pyear,06')∨
(oname,TAU)∧(aname,Tova M.)∧ (ptitle,Monitoring..)∧(cname,VLDB)∧(pyear,07')∨
(oname,TAU)∧(aname,Slava N.)∧(ptitle,OASSIS...)∧(cname, SIGMOD)∧(pyear,14')∨
(oname,TAU)∧(aname,Tova M.)∧(ptitle,A sample...)∧(cname,SIGMOD)∧(pyear,14')∨
(oname,UPENN)∧(aname,Susan D.)∧(ptitle,OASSIS...)∧(cname,SIGMOD)∧(pyear,14')
```

Figure 2: Value-level Provenance

oid	oname
1	UPENN
2	TAU

Rel. org

aid	aname	oid
3	Susan D.	1
4	Tova M.	2
5	Slava N.	2

Rel. $author$

wid	cid	ptitle	pyear
6	10	"OASSIS..."	2014
7	10	"A sample..."	2014
8	11	"Monitoring..."	2007
9	11	"Querying..."	2006

Rel. pub

aid	wid
4	6
3	6
5	6
4	7
4	8
4	9

Rel. $writes$

cid	cname
10	SIGMOD
11	VLDB

Rel. $conf$

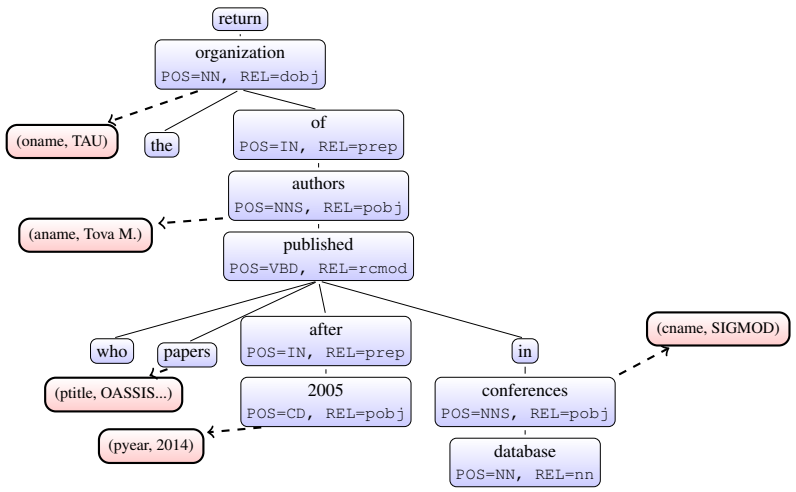
cid	did
10	18
11	18

Rel. $domainConf$

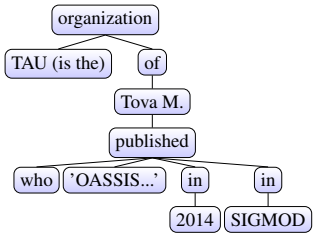
did	name
18	Databases

Rel. $domain$

Table 1: DB Instance



(a) Query Tree

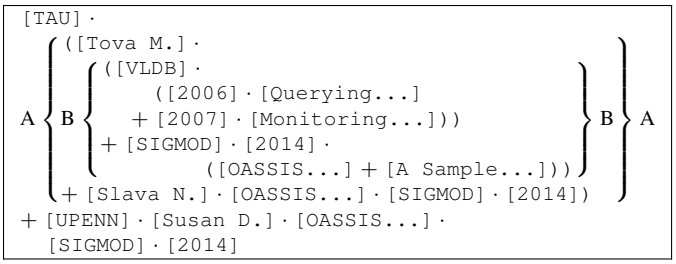


(b) Answer Tree

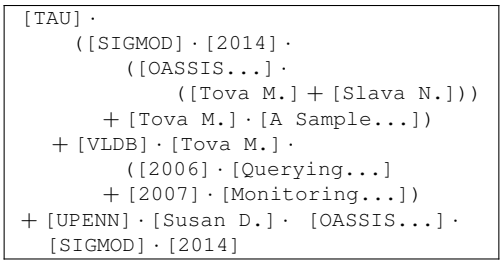
Figure 3: Question and Answer Trees

ment, and the atoms are pairs of the form (var, val) so that var is assigned val in the particular assignment. We only record variables to which a query word was mapped (these are the relevant variables for formulating the answer). For instance, the node "organization" was mapped to the variable $oname$ which was assigned the values "TAU" and "UPENN". If we were to replace the value in the organization node by the value "TAU" mapped to it, the word "organization" would not appear in the answer although it is needed to produce a coherent sentence such as the one depicted in Figure 1c. In the absence of this word, it is unclear how to connect "Tova M." and "TAU". To this end, we rely on the information encoded in the dependency tree to convert it to an answer tree based on the relationships and part-of-speech of the words in the sentence. Finally, the conversion of the answer tree in Figure 3b to a sentence is done by replacing the words of the NL query with the values mapped to them, e.g., the word "authors" in the NL query (Figure 1a) is replaced by "Tova M." and the word "papers" is replaced by "OASSIS...". The word "organization" is not replaced (as it remains in the answer tree) but rather the words "TAU is the" are added prior to it, using the part-of-speech and relationship of the word. Completing this process, we obtain the answer shown in Figure 1c.

Factorization A second key idea is related to the provenance size. In typical scenarios, a single answer may have multiple explanations (multiple authors, papers, venues and years in our example). A naïve solution is



(a) f_1



(b) f_2

Figure 4: Provenance Factorizations

to formulate and present a separate sentence corresponding to each explanation. The result will however be, in many cases, very long and repetitive. As observed already in previous work [11, 31], different assignments (explanations) may have significant parts in common, and this can be leveraged in a *factorization* that groups together multiple occurrences. In our example, we can e.g. factorize explanations based on author, paper name, conference name or year. Importantly, we impose a novel constraint on the factorizations that we look for (which we call *compatibility*), intuitively capturing that their structure is consistent with a partial order defined by the parse tree of the question. This constraint is helpful in translating the factorization back to an NL answer whose structure is similar to that of the question; again, we refer the reader to [14] for details and only show an example.

Example 2: Re-consider the provenance expression in Figure 2. Two possible factorizations are shown in Figure 4, keeping only the values and omitting the variable names for brevity (ignore the A,B brackets for now). In both cases, the idea is to avoid repetitions in the provenance expression, by taking out a common factor that appears in multiple summands. Different choices of which common factor to take out lead to different factorizations.

Consider factorization f_2 from Figure 4. "TAU" should be at the beginning of the sentence and followed by the conference names "SIGMOD" and "VLDB". The second and third layers of f_2 are composed of author names ("Tova M.", "Slava N."), paper titles ("OASSIS", "A sample...", "Monitoring...") and publication years (2007, 2014). Changing the original order of the words such that the conference name "SIGMOD" and the publication year "2014" will appear before "Tova M." breaks the sentence structure in a sense. It is unclear how to algorithmically translate this factorization into an NL answer, since we need to patch the broken structure by adding connecting phrases. One hypothetical option of patching f_2 and transforming it into an NL answer is depicted below. The bold parts of the sentence are not part of the factorization and it is not clear how to generate and incorporate them into the sentence algorithmically. Even if we could do so, it appears that the resulting sentence would be quite convoluted:

TAU is the organization **of authors who published in**
SIGMOD 2014
'OASSIS...' **which was published by**
Tova M. and Slava N.
and Tova M. published 'A sample...'
and Tova M. published in VLDB
'Querying...' in 2014
and 'Monitoring...' in 2007.
UPENN is the organization of Susan D. who published
'OASSIS...' in SIGMOD in 2014

Observe that the resulting sentence is not clear, even though it was obtained from a shorter factorization f_2; the intuitive reason is that the structure of f_2 is very different from that of the NL question, and thus is not guaranteed to admit a structure that is coherent in Natural Language. Interestingly, the sentence we would obtain in such a

way also has an edit distance from the question [18] that is shorter than that of our answer, demonstrating that edit distance is not an adequate measure here.

Instead, we propose using the structure of the original NL query to construct a factorization that, while reducing the size of the original expression, maintains the hierarchy of the words in the dependency tree. f_1 in Figure 4 maintains the word hierarchy implied by the dependency tree (Figure 3a), and thus can be converted into an answer tree and the sentence:

```
TAU is the organization of
    Tova M. who published
        in VLDB
           'Querying...' in 2006 and
           'Monitoring...' in 2007
        and in SIGMOD in 2014
            'OASSIS...' and  'A sample...'
    and Slava N. who published
        'OASSIS...' in SIGMOD in 2014.
UPENN is the organization of Susan D. who published
'OASSIS...' in SIGMOD in 2014.
```

Summarization We *summarize* explanations by replacing details of different parts of the explanation by their synopsis, e.g. presenting only the number of papers published by each author, the number of authors, or the overall number of papers published by authors of each organization. Such summarizations incur by nature a loss of information but are typically much more concise and easier for users to follow. Here again, while provenance summarization has been studied before (e.g. [2, 34]), the desiderata of a summarization needed for NL sentence generation are different, rendering previous solutions inapplicable here. We observe a tight correspondence between factorization and summarization: every factorization gives rise to multiple possible summarizations, each obtained by counting the number of sub-explanations that are "factorized together".

(A) [TAU] · *Size*([Tova M.],[Slava N.]) · *Size*([VLDB],[SIGMOD]) ·
Size([Querying...],[Monitoring...],
[OASSIS...],[A Sample...]) · *Range*([2006],[2007],[2014])
(B) [TAU]·(
[Tova M.] ·
Size([VLDB],[SIGMOD]) ·
Size([Querying...],[Monitoring...],
[OASSIS...],[A Sample...]) · *Range*([2006],[2007],[2014])
[Slava N.] · [OASSIS...] · [SIGMOD] · [2014])

(a) Summarized Factorizations

(A) *"TAU is the organization of 2 authors who published 4 papers in 2 conferences in 2006 - 2014"*

(B) *"TAU is the organization of Tova M. who published 4 papers in 2 conferences in 2006 - 2014 and Slava N. who published 'OASSIS...' in SIGMOD in 2014"*

(b) Summarized Sentences

Example 3: Reconsider Example 2; if there are many authors from TAU then even the compact representation of the result could be very long. In such cases we need to summarize the provenance in some way that will preserve the "essence" of all assignments without actually specifying them, for instance by providing only the number of authors/papers for each institution.

Re-consider the factorization f_1 from Figure 4. We can summarize it in multiple levels: the highest level of authors (summarization "A"), or the level of papers for each particular author (summarization "B"), or the level of conferences, etc. Note that if we choose to summarize at some level, we must summarize all levels below it (e.g. if we summarize for "Tova M." at the level of conferences, we cannot specify the papers titles and publication years).

Figure 5a presents the summarizations of sub-trees for the "TAU" answer, where "size" is a summarization operator that counts the number of distinct values and "range" is an operator over numeric values, summarizing them as their range. The summarized factorizations are further converted to NL sentences which are shown in Figure 5b. Summarizing at a higher level results in a shorter but less detailed summarization.

Our work in [14] is restricted to Conjunctive Queries, and extensions to more expressive forms of queries are the subject of an ongoing investigation. We next turn to briefly describe another aspect of explanations, namely explaining non-answers.

Why-not Explanations Our work so far has focused on explaining query answers in NL; an equally important type of insight that users may wish to gain concerns expected answers that do not appear in the query result set. Such insight may be useful for identifying errors in the inferred query as well as omissions in the database.

Provenance for non-answers has been extensively explored (e.g. [5–7, 10, 22]), but similarly to the positive case, a direct use of such models is unsuitable for non-experts. For instance, the work of [10] defines why-not provenance in terms of the query operators; the non-expert may be unfamiliar with formal query languages. Similarly, the work of [5] defines a notion of provenance polynomials, which may be too complex for presentation to non-experts, etc.

Again, a plausible approach (that we follow in our work-in-progress) is to leverage the structure of the NL query for presentation of why-not provenance. For example, if we leverage the work of [10] to identify "picky" operators in the query (i.e. parts of the query that were responsible for the tuple omission), then in some cases we may further track the words in the NL query that have led to the generation of each operator. Then, we can present why-not provenance by *highlighting* these words. Another approach is to suggest various alternative questions which are in the same spirit of the user's question, with a strike-through line over the word mapped to the picky operator.

Example 4: Reconsider our running example and assume that all "Hogwarts" authors who published in database conferences have done so before 2005, so all tuples that contain "Hogwarts" do not qualify due to the selection operator $pyear > 2005$. Instead of showing the selection operator itself, we can highlight the words that were mapped to this operator: "after 2005". This will give the user an idea of what has to change in the query in order to get "Hogwarts" in the result set. Another scenario may be that no author associated to "Hogwarts" has published in a database conference. In that case, we could highlight the words "database conferences" in the original NL query. Alternatively, we could suggest the question "Return the organization of authors who published papers in database conferences after 2005".

These approaches work fairly well for operators that have words in the NL query directly mapped to them. This is not always the case. For instance, for the query of Figure 1b, the triple join operator used to connect table *author* with the table *pub* through the connecting table *writes* is not specifically mentioned in the NL query, and was inferred automatically by the NL interface. Handling such cases requires further solutions whose investigation is left for future work.

3 Explanations for Query-By-Example Frameworks

Query-by-example interfaces have been extensively studied [8, 30, 44] to allow non-expert users to query a database without requiring extensive technical knowledge. These interfaces allow users to specify output examples of the desired query and then try to automatically infer it. The challenge is naturally that the size of the search space, i.e. the number of queries that are consistent with the given examples, is typically very large. Existing solutions aim at addressing this via an interactive process; for instance, users are shown at each step positive and negative examples based on generated candidate queries, and their feedback is used to focus on

a subset of these queries [8]. We claim that provenance may be very helpful in narrowing the search space in query-by-example frameworks, and may be used for this purpose in two different manners, as follows.

Query-by-provenance In [15, 16] we have proposed a framework for the inference of queries from output examples *and their explanations*. The idea is that explanations that are attached to examples may be viewed as their provenance with respect to the (unknown) ground truth query (we show in [16] an interface that allows non-experts to provide explanations).

Example 5: Consider a database of academic authors and their papers, and also consider an intended query that should return all pairs of authors who wrote a common paper. Examples of the output, i.e. pairs of authors, may share characteristics such as country of residence. This in turn may lead to the inference of a query that is very different from the intended one. In contrast, if the user also provides a co-authored paper as an explanation, this may potentially greatly restrict the search space.

This leads to three concrete questions: (1) how do we formalize the consistency of a query with examples and explanations? (2) How many different queries are consistent with a set of examples and their explanations/ (3) What is the complexity of finding a consistent query and/or enumerating all of them? In [15] we provide initial answers to these questions. We formally define a generic notion of query-by-provenance that applies to any *provenance semiring* [21]. We show that the answers to the questions (2) and (3) above greatly depend on the choice of semiring. For instance, the number of queries for a given provenance polynomial (NX) may be exponential in the sum of the arities of the relations participating in a monomial. On the other hand, if the provenance is given in the Why(X) semiring (corresponding to the why-provenance of [9]), there may be infinitely many consistent queries. Yet, we show in [15] a "small-world" property, namely that there exists a consistent query whose size is polynomial in the number of attributes in the output example, the number of distinct relation names in the provenance, and the largest cardinality of a set in the provenance.

Using Provenance to Choose Between Candidate Queries As mentioned above, even in the presence of explanations, there may be a large number of consistent candidate queries, and further user feedback is required to choose a "correct" one, i.e. one that matches their intention. Provenance may be highly useful in this respect as well; it is common practice to procure feedback for possible answers that would allow differentiating between candidate queries (i.e. ask about the correctness of answers of one query that are non-answers of another one). But it is not always trivial for users to state whether or not a proposed result should appear in the output; having the system explain the rationale for computing this answer (i.e. present its provenance with respect to the candidate query being considered), can be highly useful.

Example 6: Consider an ontology of authors and papers, and a query asking for all authors with Erdős number 2. Examples for the query output, i.e. example authors, will likely be un-indicative of the actual intended query, since the authors may share many other characteristics. The *provenance* of an example author with respect to the intended query will be one of her co-authorship paths to Erdős of length 2, which reveals much more information on the inferred query and allows for its validation by the user.

Following our general principle, both the presentation of provenance and the procurement of explanations take a form that follows the standard interaction of the user with the system. For instance, users attach explanations to examples that they would give regardless of provenance, and so they can use the rationale they have already employed in choosing the examples, to form the explanation. Provenance for a sample answer produced by a candidate query can be shown by e.g. the relevant path in an RDF setting (in our ongoing work), and the candidate answers themselves should be carefully chosen to "resemble" the user-provided examples. Employing these principles allows for bridging the gap between formal provenance models and non-expert users.

4 Machine Learning Results Explanation

The two example domains shown above are ones where the underlying models are based on some query language for which provenance may be tracked. Can we leverage the developed ideas for additional settings? In this section we briefly describe our work-in-progress, exploring the possibility of generating explanations for non-experts that shed light on the result of Machine Learning (ML) models.

Despite its wide adoption, one drawback of ML is the complexity level of its models which makes it difficult to understand the reasons for a given result. Such understanding is crucial to assess (and potentially improve) the quality of a model and identify errors in its output. Recognizing this need, there is a large body of work on understandable Machine Learning. One approach in this respect is to explain the model as a whole, e.g. by approximating its specification through a simpler model, such as rules (see e.g. [23]). A second approach is not to explain the model as a whole but rather to present the reasoning underlying individual predictions [35]. This latter approach is close in spirit to provenance, and will be the focus of our short discussion.

We start with a simple model of explanations based on minimal changes, and gradually refine it.

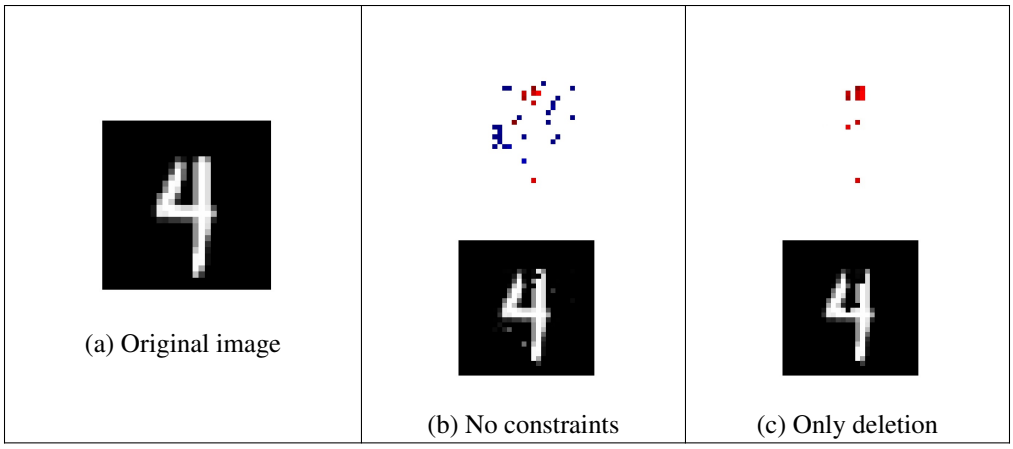
Example 7: Consider a simple linear regression model for loan applications, where the model takes into account two binary parameters, $income$ and $debt$. For each loan request, the model will return a score based on the following formula $y = 0.5 + \frac{income}{2} - \frac{debt}{2}$. Assume that loans are approved only if they have model score of 1. Further consider an applicant with $x = \{income : 1, debt : 1\}$. The model returns the score of 0.5, and the application is consequently denied. Naturally, the applicant may be interested in understanding the denial reason, and may ask e.g. **(1)** what factors have influenced the decision? and **(2)** what changes in the application may lead to its acceptance next time?

A possible explanation model, in the spirit of why-not provenance, is that of *minimal changes* to the data that would change the model decision. In this example, to answer the second question, we may look for minimal changes that lead to loan acceptance (in our example, this involves changing the profile to $x = \{income : 1, debt : 0\}$). This also partly answers the first question, as we understand that the $debt$ parameter has a major influence on the model result. We may further look for changes that would cause to model the *decrease* its confidence in the application. By doing so, we would find out that changing the income value such that $x = \{income : 0, debt : 1\}$ would reduce the model score to 0. Combined, we observe that both $income$ and $debt$ parameters have a significant influence on the model decision.

In the context of database queries the related problems of explanations using minimal changes has been studied in [27, 28, 43]. However, in the domain of machine learning the subject of minimal modifications in the input that change the model output has been studied in a different context than that of explanations, namely adversarial examples (see e.g. [20, 41]) where the goal is to "fool" the model. For explanations, we may need a more refined model: in our example, it does not make sense to recommend the applicant to change features she can not control such as her age. Hence, we may be interested in the minimal changes that lead to the satisfaction of some pre-specified constraints.

Example 8: Figure 6a presents an image of handwritten 4 that was tagged as 9 by a simple Random Forest model which we have trained using the MNIST data set. Two examples for minimal changes to the image that lead to a decision of "4" are shown in Figures 6b and 6c. In Figure 6b there were no constraints on the allowed modifications, and we can see that many pixels have changed. Some of them are not even close to the digit, and do not serve as a convincing explanation. In Figure 6c we present the result of the minimal sequence of *pixel deletions* (the constraint is that only deletions are allowed) that lead to the tag "4". We can see that most of the deletion were of pixels in the top of the image, providing further insight.

Can we gain further insights by *summarizing* explanations for multiple instances that share some property? Our initial results indicate that this is the case.



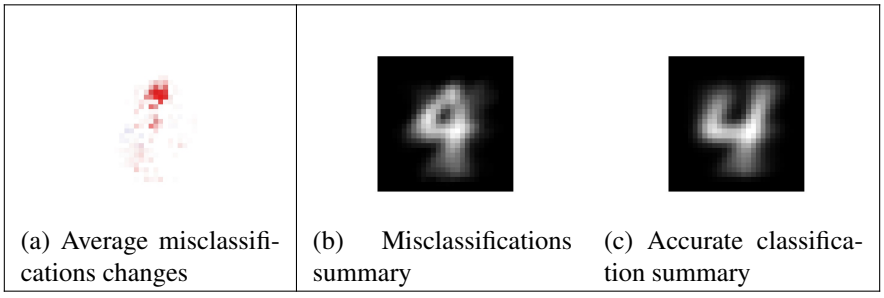
(a) Original image

(b) No constraints

(c) Only deletion

Figure 6: Change constraints

Example 9: In Figure 7a we depict the *average* changes over multiple instances that were tagged "9" but for which the ground truth tag is "4". Now we can clearly see that deleting the top pixels of the digit is a general "fix", and so their existence is a valid explanation for the misclassification. We can verify the correctness of this insight by Figures 7b and 7c that present the average of 4 images that were tagged as 9 and 4 respectively by our model. Observe that the upper part of the digits classified as 9 is closed and that of digits classified as 4 is open. The gained insight is that the model learned how to recognize images of 4 with open top, but images of 4 with close top are misclassified as 9.



(a) Average misclassifications changes

(b) Misclassifications summary

(c) Accurate classification summary

Figure 7: Summarized explanations

5 Conclusion

We have discussed the usefulness of provenance presentation to non-experts, and have demonstrated it in three contexts: NL database interfaces, query-by-example frameworks, and Machine Learning. We have further described preliminary solutions for transforming provenance into user-friendly explanations, highlighting some common principles. Further scaling up the solutions, applying them to additional frameworks and applications, and improving the clarity of explanations are all subjects of our ongoing investigation.

Acknowledgments This research has been partially supported by the Israeli Science Foundation (ISF), by ICRC - The Blavatnik Interdisciplinary Cyber Research Center and by Intel. The different contributions of Nave Frost and Amir Gilad are each part of their respective PhD research conducted at Tel Aviv University.

References

[1] A. Abouzied, J. M. Hellerstein, and A. Silberschatz. Playful query specification with dataplay. *PVLDB*, 5(12), 2012.

[2] E. Ainy, P. Bourhis, S. B. Davidson, D. Deutch, and T. Milo. Approximated summarization of data provenance. In *CIKM*, 2015.

[3] Y. Amsterdamer, A. Kukliansky, and T. Milo. A natural language interface for querying general and individual knowledge. *PVLDB*, 8(12), 2015.

[4] N. Bakibayev, D. Olteanu, and J. Zavodny. FDB: A query engine for factorised relational databases. *PVLDB*, 5(11), 2012.

[5] N. Bidoit, M. Herschel, and A. Tzompanaki. Efficient computation of polynomial explanations of why-not questions. In *CIKM*, 2015.

[6] N. Bidoit, M. Herschel, and K. Tzompanaki. Immutably answering why-not questions for equivalent conjunctive queries. In *TaPP*, 2014.

[7] N. Bidoit, M. Herschel, and K. Tzompanaki. Query-based why-not provenance with nedexplain. In *EDBT*, 2014.

[8] A. Bonifati, R. Ciucanu, and S. Staworko. Interactive inference of join queries. In *EDBT*, 2014.

[9] P. Buneman, S. Khanna, and W. C. Tan. Why and where: A characterization of data provenance. In *ICDT*, 2001.

[10] A. Chapman and H. V. Jagadish. Why not? In *SIGMOD*, 2009.

[11] A. P. Chapman, H. V. Jagadish, and P. Ramanan. Efficient provenance storage. In *SIGMOD*, 2008.

[12] A. Das Sarma, A. Parameswaran, H. Garcia-Molina, and J. Widom. Synthesizing view definitions from data. ICDT, 2010.

[13] D. Deutch, N. Frost, and A. Gilad. Nlprov: Natural language provenance (demo). *PVLDB*, 9(13), 2016.

[14] D. Deutch, N. Frost, and A. Gilad. Provenance for natural language queries. *PVLDB*, 10(5), 2017.

[15] D. Deutch and A. Gilad. Learning queries from examples and their explanations. *CoRR*, 2016.

[16] D. Deutch and A. Gilad. Qplain: Query by explanation (demo). In *ICDE*, 2016.

[17] D. Deutch, A. Gilad, and Y. Moskovitch. Selective provenance for datalog programs using top-k queries. *PVLDB*, 8(12), 2015.

[18] M. Emms. Variants of tree similarity in a question answering task. In *LD*, 2006.

[19] I. Foster, J. Vockler, M. Wilde, and A. Zhao. Chimera: A virtual data system for representing, querying, and automating data derivation. *SSDBM*, 2002.

[20] I. J. Goodfellow, J. Shlens, and C. Szegedy. Explaining and Harnessing Adversarial Examples. *ArXiv e-prints*, 2014.

[21] T. Green, G. Karvounarakis, and V. Tannen. Provenance semirings. In *PODS*, 2007.

[22] M. Herschel. A hybrid approach to answering why-not questions on relational query results. *J. Data and Information Quality*, 5(3), 2015.

[23] J. Huysmans, B. Baesens, and J. Vanthienen. Using rule extraction to improve the comprehensibility of predictive models. 2006.

[24] G. Karvounarakis, Z. G. Ives, and V. Tannen. Querying data provenance. In *SIGMOD*, 2010.

[25] D. Küpper, M. Storbel, and D. Rösner. Nauda: A cooperative natural language interface to relational databases. SIGMOD, 1993.

[26] F. Li and H. V. Jagadish. Constructing an interactive natural language interface for relational databases. *PVLDB*, 8(1), 2014.

[27] A. Meliou, W. Gatterbauer, J. Y. Halpern, C. Koch, K. F. Moore, and D. Suciu. Causality in databases. *IEEE Data Eng. Bull.*, 2010.

[28] A. Meliou, W. Gatterbauer, K. F. Moore, and D. Suciu. The complexity of causality and responsibility for query answers and non-answers. *Proceedings of the VLDB Endowment*, 2010.

[29] D. Mottin, M. Lissandrini, Y. Velegrakis, and T. Palpanas. Exemplar queries: Give me an example of what you need. In *VLDB*, 2014.

[30] D. Mottin, M. Lissandrini, Y. Velegrakis, and T. Palpanas. Exemplar queries: Give me an example of what you need. *PVLDB*, 7(5), 2014.

[31] D. Olteanu and J. Závodný. Factorised representations of query results: Size bounds and readability. ICDT, 2012.

[32] A.-M. Popescu, O. Etzioni, and H. Kautz. Towards a theory of natural language interfaces to databases. In *IUI*, 2003.

[33] F. Psallidas, B. Ding, K. Chakrabarti, and S. Chaudhuri. S4: Top-k spreadsheet-style search for query discovery. SIGMOD, 2015.

[34] C. Ré and D. Suciu. Approximate lineage for probabilistic databases. *PVLDB*, 1(1), 2008.

[35] M. T. Ribeiro, S. Singh, and C. Guestrin. "why should i trust you?": Explaining the predictions of any classifier. KDD, 2016.

[36] T. Sellam and M. L. Kersten. Meet charles, big data query advisor. CIDR, 2013.

[37] Y. Shen, K. Chakrabarti, S. Chaudhuri, B. Ding, and L. Novik. Discovering queries based on example tuples. In *SIGMOD*, 2014.

[38] Y. L. Simmhan, B. Plale, and D. Gannon. Karma2: Provenance management for data-driven workflows. *Int. J. Web Service Res.*, 2008.

[39] D. Song, F. Schilder, and C. Smiley. Natural language question answering and analytics for diverse and interlinked datasets. NAACL, 2015.

[40] D. Song, F. Schilder, C. Smiley, C. Brew, T. Zielund, H. Bretz, R. Martin, C. Dale, J. Duprey, T. Miller, and J. Harrison. TR discover: A natural language interface for querying and analyzing interlinked datasets. In *ISWC*, 2015.

[41] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. J. Goodfellow, and R. Fergus. Intriguing properties of neural networks. *CoRR*, 2013.

[42] Q. T. Tran, C.-Y. Chan, and S. Parthasarathy. Query by output. In *SIGMOD*, 2009.

[43] E. Wu and S. Madden. Scorpion: Explaining away outliers in aggregate queries. *Proceedings of the VLDB Endowment*, 2013.

[44] M. Zhang, H. Elmeleegy, C. M. Procopiuc, and D. Srivastava. Reverse engineering complex join queries. In *SIGMOD*, 2013.

[45] M. M. Zloof. Query by example. In *AFIPS NCC*, 1975.