Towards Automating Data Narratives

Yolanda Gil
Information Sciences Institute
University of Southern California
Los Angeles, CA, USA
gil@isi.edu

Daniel Garijo
Information Sciences Institute
University of Southern California
Los Angeles, CA, USA
dgarijo@isi.edu

ABSTRACT

We propose a new area of research on automating data narratives. Data narratives are containers of information about computationally generated research findings. They have three major components: 1) A record of events, that describe a new result through a workflow and/or provenance of all the computations executed; 2) Persistent entries for key entities involved for data, software versions, and workflows; 3) A set of narrative accounts that are automatically generated human-consumable renderings of the record and entities and can be included in a paper. Different narrative accounts can be used for different audiences with different content and details, based on the level of interest or expertise of the reader. Data narratives can make science more transparent and reproducible, because they ensure that the text description of the computational experiment reflects with high fidelity what was actually done. Data narratives can be incorporated in papers, either in the methods section or as supplementary materials. We introduce DANA, a prototype that illustrates how to generate data narratives automatically, and describe the information it uses from the computational records. We also present a formative evaluation of our approach and discuss potential uses of automated data narratives.

Author Keywords

Reproducibility; computational workflows; semantic workflows; WINGS; data narratives; explanation

ACM Classification Keywords

H.5.3. Information interfaces and presentation (e.g., HCI): Group and organization interfaces.

INTRODUCTION

The reproducibility of published scientific research has received significant attention [49]. For computational experiments, published papers often provide insufficient information about the data, protocols, software, and overall method used to obtain the new results [54]. A major challenge in reproducibility is the lack of appropriate support for authors to capture exactly how experiments were performed. Once the work is finished, authors write an account in their articles by retrospective reconstruction of the work that was actually done, relying on their memory and notes kept along the way. For research carried out in the field or laboratory, tracking accurately what was done can require discipline and effort. But why should reproducibility be challenging for computational experiments, when we have the infrastructure to capture accurately the computations that were carried out? We should have automated tools that ensure that the descriptions that are written about computational experiments are in fact accurate and provide enough detail for transparency and reproducibility.

In this paper, we propose data narratives as a new approach to automatically generate text to describe computational analyses. This textual description of methods is human readable and is automatically generated from provenance records, workflows, and other formal records of computational experiments. We ensure that they offer an inspectable description of the computations by requiring that all the evidence mentioned is persistently stored. They are guaranteed to reflect the actual computations carried out, and are human-readable so they can be included in papers if desired. Because they are automatically generated, the narratives can be easily customized to the reader's level of expertise and interest.

The paper begins motivating the need to improve on the way that computational methods are described in papers, and the existing approaches to document them and improve the way articles are written. We then show examples of different textual descriptions that could be automatically generated for the same computational experiment. We discuss a range of dimensions for constructing different data narrative accounts depending on the reader's purpose and interest in detail. We propose an approach where data narratives are generated based on a combination of explanation patterns and query patterns. We describe DANA, an initial prototype that illustrates how data narratives can be implemented, and discuss the computational artifacts and the queries used to obtain information needed about the computational experiment. We present a formative evaluation of our approach regarding the understandability and usefulness of the automatically generated narratives, and conclude with a discussion of potential applications and future work.

MOTIVATION: METHODS IN PUBLISHED ARTICLES

As computational methods have become ubiquitous in scientific research, it is imperative that we examine how they are recorded and published as part of the scientific record that supports new results and claims. Computational methods are

IUI 2017, March 13-16, 2017, Limassol, Cyprus
 ISBN 978-1-4503-4348-0/17/03. $15.00
DOI: http://dx.doi.org/10.1145/3025171.3025193

complex procedures, which are challenging for humans to explain with enough detail when teaching others [26]. In this section we describe the challenges specific to describing computational methods in scientific articles.

Textual Descriptions of Methods

Textual descriptions of methods in articles may be incomplete [55; 20; 16]. Authors focus on conveying the major contributions of the work and describe the methods in that light, omitting details that may be important for transparency and reproducibility. For example, [20] describes our work to reproduce an article whose authors had published the data, software, and results. We analyzed whether the original article and supporting materials were enough to reproduce the method for three kinds of researchers: 1) someone with basic knowledge about the domain at hand, such as a developer hired to work in a lab; 2) a novice researcher, such as a new student; 3) an experienced researcher with the same level of expertise as the authors. We created reproducibility maps, that showed whether each of those categories of users could figure out from the text of the paper how the work was done. Even in this case where the original authors set out to make their work as transparent as possible, the reproducibility maps showed that only experienced researchers were able to figure out how to fully reproduce the work. There are many similar results in the literature, some mentioning the lack of publication of data [55] and others the lack of details in the description of methods:

> "Data processing is often not described well enough to allow for exact reproduction of the results, leading to exercises in 'forensic bioinformatics' where aspects of raw data and reported results are used to infer what methods must have been employed." [3].

There are several reasons why text descriptions of methods are riddled with problems. First, articles often have space limitations, so authors tend to omit anything that seems not important. Second, they are manually written without any particular guidance, it is easy for authors to provide imprecise descriptions. Finally, computational methods are often complex procedures with non-linear structures that are hard to describe with the sequential nature of text [15; 53].

Even when authors endeavor to describe enough details, textual descriptions are often ambiguous. A study reported in [37] looked at writing software from scratch based on the textual descriptions reported in geophysics papers. As the authors state:

> "Ambiguity in program descriptions leads to the possibility, if not the certainty, that a given natural language description can be converted into computer code in various ways, each of which may lead to different numerical outcomes. […] Ambiguity can occur at the lexical, syntactic or semantic level and is not necessarily the result of incompetence or bad practice. It is a natural consequence of using natural language and is unavoidable." [37]

We also find that the methods sections of articles mix general methods with specific details of the executions carried out. The original article of the reproducibility work that we mentioned above [20] provides a good example as it describes how to match proteins and drugs:

> "The SMAP software was used to compare the binding sites of the 749 M.tb protein structures plus 1,446 homology models (a total of 2,195 protein structures) with the 962 binding sites of 274 approved drugs, in an all-against-all manner. While the binding sites of the approved drugs were already defined by the bound ligand, the entire protein surface of each of the 2,195 M.tb protein structures was scanned in order to identify alternative binding sites. For each pairwise comparison, a p-value representing the significance of the binding site similarity was calculated." [38]

Note that this paragraph mixes high-level abstract information with execution details: it mentions the particular software (SMAP) as well as its function (comparing binding sites), the size of the specific input datasets (962 and 274), and the general type of result returned (a p-value).

Although there are many tools and recommendations of best practices for authors [33], it is up to the scientists to decide what to include in an article and its methods section.

In summary, textual descriptions of methods in articles are far from ideal, since the text tends to be: 1) Incomplete, omitting important details about the computations performed; 2) Ambiguous, having several interpretations of how the computations were actually done; 3) Blended, combining general methods with execution details.

Executable Papers and Electronic Notebooks

Another area that has received significant attention is papers that go beyond a static PDF [7; 8; 50]. In some cases, articles are augmented so that figures that show visualizations can be interactive. In other cases, the references are explicitly linked to the papers cited, so it is easy to navigate the related work. Although these efforts enhance traditional paper narratives, the closest to our goals is the work on enhancing the textual descriptions of methods.

Interactive notebooks are gaining popularity, including Jupyter Notebook for Python [39], knitr for R [58] and the Computable Document Format for Mathematica [57]. Executable papers are designed to enable readers to inspect and re-run experiments [47]. Readers can inspect the data and the code in these frameworks, use them to re-generate figures, and try different data and parameters to explore variations on the experiments that were done. However, the text in these notebooks is completely generated by hand. Therefore, there is no guarantee that the text reflects the actual computations. In addition, much like with a traditional paper there is a single textual description that is supposed to satisfy all readers no matter their level interest or level of expertise.

Descriptions of Workflows

Workflows capture a computational analysis as a dataflow among steps. Workflow repositories like myExperiment [13] and CrowdLabs [44] provide mechanisms to publish and search workflows, particularly to improve reproducibility and sharing of computational experiments. However, the

descriptions of workflows are manually generated and therefore are as incomplete as those in scientific articles.

In prior work we analyzed the textual descriptions of workflows from the myExperiment repository [34]. We found significant differences between what was included in the textual descriptions and the actual formal specification of the workflows.

Workflows mix major method steps with ancillary steps that do for example minor data reformatting. In other work we analyzed workflows to identify by hand general categories of steps (*motifs*) that make such distinctions [21]. But workflows themselves have no explicit mention of the relative importance of steps and all steps are treated equally.

In summary, although workflows provide a formal computational representation of methods, the descriptions that can be generated for workflows alone are: 1) Incomplete, where workflow representations cannot express important semantic properties of steps; 2) Flat, with abstractions often absent from the workflow structure; and 3) Undifferentiated, as there is no explicit distinction between important steps and ancillary steps.

Workflows can be integral elements of data narratives, but need to be appropriately explained and presented to scientists in order to improve understandability and increase reuse.

DATA NARRATIVES

Since it is possible to automatically record in detail the computational methods of papers, it is curious that their descriptions are still written manually rather than generated automatically. The key idea of data narratives is that a system can not only keep detailed provenance records of how an analysis was done, but also generate automatically human-readable descriptions of those records that can be presented to users and ultimately included in papers or reports.

We pose the challenge of automatically generating text descriptions of computational methods used to generate the data being described. Several accounts could be generated from the same computational experiment. They should be linked to the provenance records and to the individual datasets and software used in the analysis. Data narratives provide a human-understandable report of the computations done to generate new data, grounded on inspectable evidence needed for reproducibility.

A *data narrative* includes: 1) interlinked provenance records, high-level workflows, and other relevant information about a computational experiment, 2) persistent identifiers for all the data, software, and workflows used in the analysis, and 3) *data narrative accounts* that contain alternative descriptions of an analysis with a different focus or level of detail. Data narratives must be:

- *Human-readable*, unlike traditional traces and provenance records that are for machine processing rather than for human consumption.
- *Customizable*, where different narrative accounts could be generated for readers with different purposes and expertise levels.

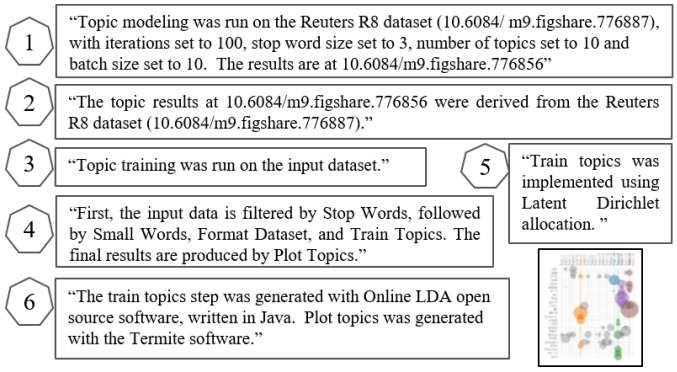

Figure 1: A range of examples of data narrative accounts for a dataset, the topic model shown at the bottom right. The different data narratives explain from different perspectives and various levels of abstraction how the dataset was generated.

- *Persistent*, since the evidence that they portray is always permanently available as back up.

Because they are automatically generated, data narratives should also be:

- *Accurate*, reflecting exactly the computations that took place and being true to the actual computations run.
- *Inspectable*, interlinking information to make the computational analysis understandable, verifiable, and reproducible.
- *Publishable*, where computer-generated reports would become part of papers. They could be included in the methods section of the paper or as supplementary materials, and complemented by additional context provided manually by the authors.

Figure 1 illustrates with examples a wide range of data narrative accounts for the topic model visualization at the bottom right of the figure. The dataset to be described is a topic model as a term-topic matrix generated with a Latent Dirichlet Allocation (LDA) algorithm [6] and visualized with Termite [11]. Data narrative account (1) treats the method itself as a black box, describing input datasets, execution parameters, and results showing persistent identifiers for input and output data. Account (2) describes the main inputs datasets that were involved in the creation of the target results. Account (3) describes the main steps of the method based on their functionality. Data narrative account (4) describes all the steps. Account (5) gives a more specific description of the algorithm used. Data narrative account (6) gives a description focused on the software used for the computations. This small example illustrates the different data narrative accounts that could be generated to describe the same dataset. Our goal is to generate these different kinds of data narrative accounts automatically.

GENERATING DATA NARRATIVE ACCOUNTS

Our approach to generate data narrative accounts is to combine pattern-based explanations with query-based information retrieval over the computational records.

Explanation Patterns

We follow natural language generation techniques based on explanation patterns [12]. Explanation patterns contain the main text to include and a few variables that are filled by objects, in our case resulting from queries. The following are examples of explanation patterns used for data narrative accounts (1) and (4) from Figure 1 respectively:

> [P1] "?method was run on the ?dataset1Name dataset (?doi1), (…) and ?datasetNName (?doiN), with ?param1 set to ?param1Value, (…) and ?paramN set to ?paramNValue. The ?resultName results are at ?resultDOI".
>
> [P2] "First, the input data is processed by ?step1, followed by ?step2, ?step3, (…) and ?stepN. The final results are produced by ?stepN".

We have created hierarchies of explanation patterns along the following major dimensions:

- *Number of steps*: This can range from zero (where the method is treated as a black box) to all steps.
- *Experimental relevance of steps, parameters, and datasets*: This ranges from experiment-critical to experiment-relevant to ancillary.
- *Algorithmic abstractions*: These range from the specific software run, to classes of algorithms, to very high-level function descriptions.
- *Execution information*: These range from focusing on the execution (e.g., runtime of steps), to software and datasets used, to more general descriptions of the execution results.
- *Software descriptions*: These range from just the name, to high-level characteristics of the software (e.g., language, license, authors), to very detailed documentation.
- *Data descriptions*: These are based on hierarchies of data types from more general to more specific.

For each dimension and level of detail, a different explanation pattern would be used to generate the data narrative. A data narrative account could be along a single dimension. It can also combine together more than one dimension. For example, a data narrative may show only experiment-critical steps combined with very detailed software descriptions.

Query Patterns

Query patterns access information sources about the computational experiment, such as:

- Provenance records that include workflows, software components, and datasets used.
- Workflows descriptions at different levels of abstraction. These can include sub-workflow abstractions as well as abstract descriptions of steps as hierarchies of increasingly more specific classes.
- Data and software registries, where metadata is available for datasets and software components of workflows.

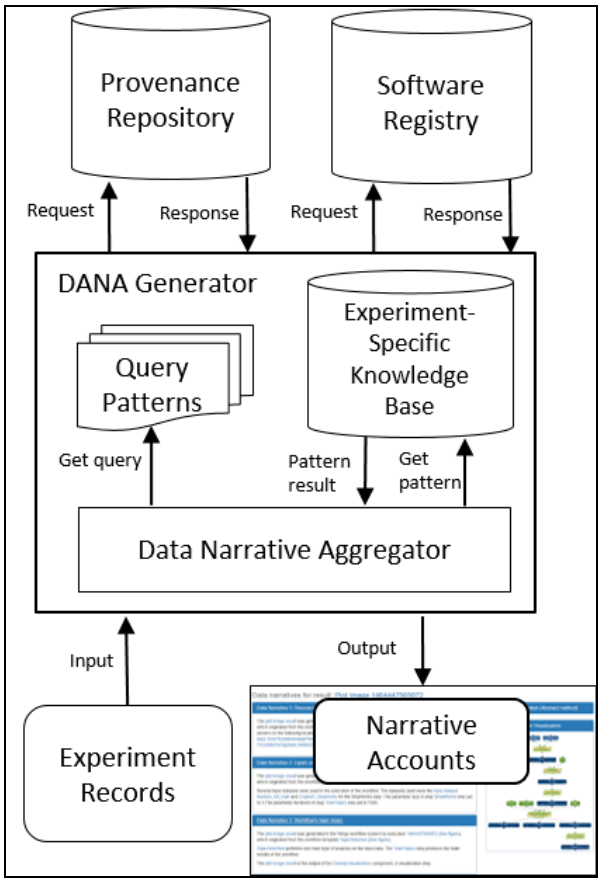

Figure 2: Overview of DANA, a prototype system for automated generation of data narratives.

Better data narrative accounts can be generated when there are appropriate descriptions and abstractions in these information sources.

DANA: AUTOMATICALLY GENERATED DATA NARRATIVES

DANA is a prototype that automatically generates data narratives using explanation patterns and query patterns. Figure 2 shows an overview of its main components. DANA takes as input the *Experiment Records*, which includes the following information:

1. A link to a provenance record in a repository that describes the execution as well as links to more abstract workflow descriptions used to generate the dataset being narrated.
2. A set of persistent identifiers for all the datasets and software in the provenance record
3. A link to a software catalog where all the software used in the workflow steps is described with basic software metadata [54]. The workflow steps must correspond to specific functions carried out by software components in this catalog. These functions are described in an ontology from specific to more general (e.g., a merge sort step is a subtype of sorting step). They are also annotated with the relative criticality that they have in an experiment so they can be included or not in more summarized data narrative accounts.

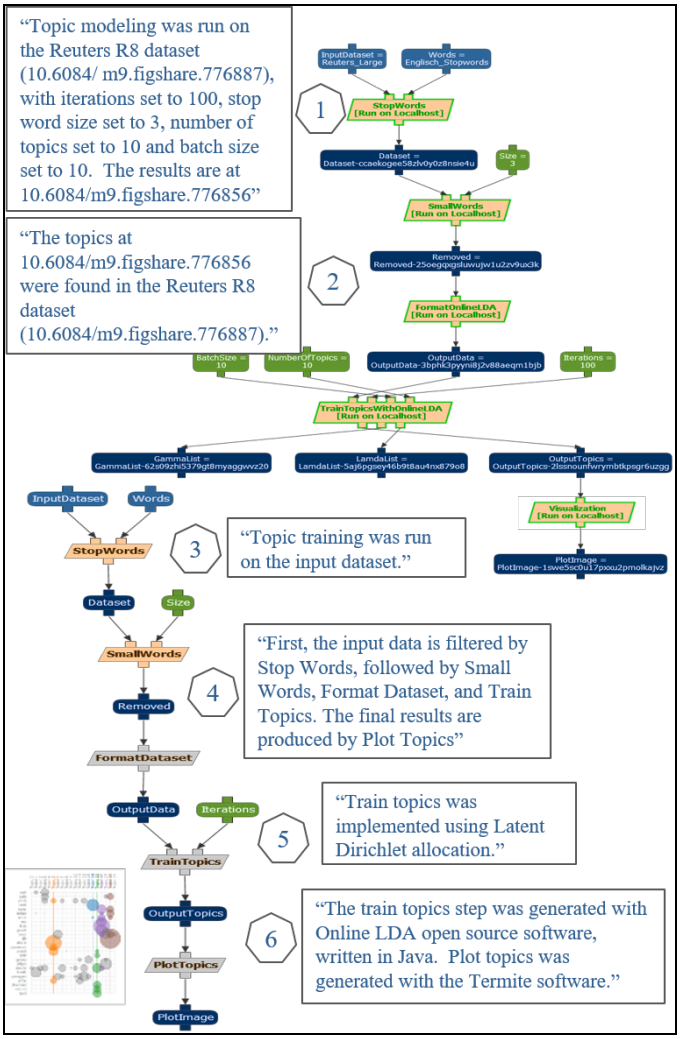

Figure 3: Overview of how the data narratives in Figure 1 can be generated from workflow execution information (shown at the top), or an abstract workflow template (shown at the bottom).

DANA contains six types of data narrative accounts that focus on different views of the computational experiment: 1) *execution view*, focused on the overall method, inputs and parameters, 2) *data view*, focused on those major inputs that directly influence the results, 3) *functionality view*, focused on the functionality of the most important steps of the analysis, 4) *dependency view*, focused on describing the general dataflow between the abstract steps of the workflow, 5) *implementation view*, focused on how the functionality of the steps was accomplished with concrete algorithms or implementations, and 6) *software view*, which shows details of the software packages used (e.g., version, license, etc.). Figure 3 shows examples of these 6 narrative types, initially introduced in Figure 1, as well as the workflow execution and template used. Each narrative type has its own explanation and query patterns.

Using the Experiment Records, the *DANA Generator* creates an *Experiment-Specific Knowledge Base (ESKB)*. The ESKB includes all the relevant information about the traces, workflow, software and annotations, linking them together. The ESKB allows DANA to reason about this information in response to the queries from the explanation patterns.

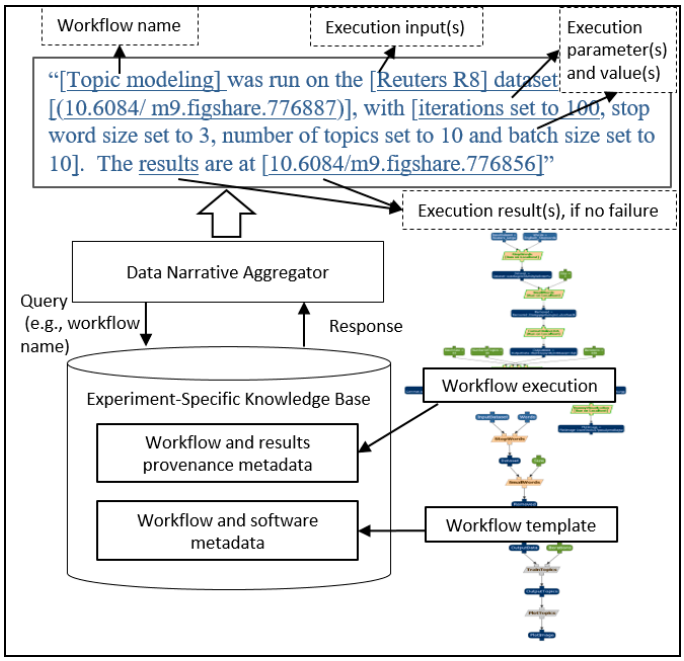

Figure 4: Creating the first data narrative shown in Figure 3 from the narrative-specific knowledge base.

Once the knowledge base is created, the DANA *Aggregator* uses a set of *query patterns* to retrieve from the knowledge base the information needed for the different narrative templates (e.g., inputs used to generate a particular result, dependencies on each of the processes of the workflow, information about the version of different software components, etc.). The Aggregator then uses the information retrieved by each query pattern in each explanation pattern to produce each of the narrative accounts. An example is depicted in Figure 4, where the DANA Aggregator completes the first narrative account presented in Figure 3 by querying the knowledge base for crucial pieces of information from the workflow template and execution provenance records.

In order to generate rich provenance records and workflows for computational experiments, DANA uses the WINGS semantic workflow system. WINGS represents high-level abstract workflow templates and uses them to generate executable workflows [28]. Figure 3 illustrates the difference, using a topic training workflow where the most popular topic trends are extracted from a collection of documents [30]. Figure 3 shows an example of a workflow template at the bottom. It has abstract steps, as it includes a step that represents the class of algorithms to train topic models, all of them have a parameter which is the number of iterations. Workflow components in WINGS are organized in an ontology, so more general classes represent more abstract descriptions of the algorithms. Figure 3 shows an example of an executable workflow at the top, with specific algorithms executed, specific datasets used, specific parameter values, and specific resources for each computation. In particular, the step for topic training is instantiated to Online LDA, which takes in additional parameters and generates additional outputs besides the topic model itself.

The component ontology has a general class of "Train topics", with a subclass "Train topics LDA" which represents Latent Dirichlet Allocation (LDA) algorithms. There are three available implementations in the component ontology: Mallet [46], Online LDA [36], and Parallel LDA [56]. The workflow execution refers to a specific implementation (Online LDA), while the higher-level template refers to a class of algorithms (Train topics). The different narratives in Figure 1 can be generated from information in each of these types of workflows and component ontologies, as shown in Figure 4.

WINGS exports the provenance of new data results using a methodology described in [17]. The provenance record contains the executed workflow as well as the associated abstract workflow. The methodology exports the workflow templates and executions with the OPMW model [18]. OPMW extends existing models like P-Plan [19] for representing workflow templates, and the W3C provenance standard PROV [42] for addressing the workflow execution provenance. OPMW supports the representations of workflows at a fine granularity with a lot of details pertaining to workflows that are not covered in those more generic languages. OPMW also allows the representation of links between a workflow template and a workflow execution that resulted from it, which we use for our narratives. All the links between a workflow and its execution are captured in the provenance repository [25], allowing DANA to retrieve any information needed to generate data narratives.

A second major source of information for DANA to generate narratives is the persistent identifiers for datasets and software in the provenance record. All the steps, inputs, outputs and intermediate results exported with the publishing methodology are assigned persistent unique uniform resource identifiers (URIs). URIs are unique for each resource, and they make them accessible by both humans and machines, as they allow returning different contents (i.e., HTML for humans and a more structured content like RDF/XML for machines). Another advantage of using URIs is that they are compatible with minting persistent digital object identifiers (DOIs) to resources when desired. DOIs are important resources for attributing credit to any scientific output. In DANA, an author can choose to mint a DOI for a dataset generated by a workflow, or to access it through its URI.

The third major source of information for narrative generation is a software catalog. The WINGS software catalog includes an ontology with abstract classes of workflow components. For software metadata, we use OntoSoft, [32] which includes an ontology of software metadata that can be queried to retrieve metadata properties about the software. The metadata captured by OntoSoft falls into six major categories based on information that a scientist would seek about the software: 1) identify software, 2) understand and assess software, 3) execute software, 4) get support for the software, 5) do research with the software, and 6) update the software. Finally, the workflow components are annotated with their criticality level according to their experimental relevance, as are their input and output datasets and parameters. For the step annotations, we use common workflow motifs [21], which are domain independent conceptual abstractions on workflow steps that aim to capture their main functionality (e.g., combine, reformat, filter, visualize, etc.). Based on this functionality, we assess how critical a step is for the workflow. For example, if a workflow motif for a step is "filtering", then it is a kind of data pre-processing and therefore not an experiment-critical step. The common workflow motifs are organized in a catalog with 23 motif types, based on the most common motifs in a large corpus of workflows from several workflow systems [21].

DANA takes the Experiment Descriptions as RDF [43]. It uses a semantic triple store [10] for the ESKB. We use SPARQL queries [51] to formulate query patterns to retrieve the information needed for the explanation patterns.

Figure 5 shows the narratives that DANA generated automatically for the dataset and workflows shown in Figure 3. An example of a query pattern used in data narrative 1 to get the location, DOI, and workflow used to generate a dataset is:

```
select ?label ?loc ?doi ?wf ?wfName where {
 <datasetID> rdfs:label ?label;
       opmo:account ?exec;
       opmw:hasLocation ?loc;
       bibo:doi ?doi.
 ?exec opmw:correspondsToTemplate ?wf.
 ?wf rdfs:label ?wfName.
}
```

In the query, the datasetID refers to the identifier of the dataset described with the narrative. The ?label is a human readable description of the dataset, while the ?loc refers to the location where the file is stored. The ?doi refers to the permanent identifier used to store the dataset and the ?exec refers to the execution that produced the dataset described in the narrative. Finally, ?wf is used to refer to the workflow that led to the execution that resulted in the dataset, along with its human readable description ?wfname. For simplicity, we use namespaces (opmw, opmo, bibo, rdfs) for referring to the properties linking the different identifiers in the knowledge base.

Data narrative 1 also uses a query pattern to extract the location (i.e., a URI or DOI) of the dataset and the values of the parameters used as input of the workflow:

```
select ?input ?loc ?value where {
 <datasetID> opmo:account ?exec.
 ?input a opmw:WorkflowExecutionArtifact.
       opmo:account ?exec.
 optional(<input> opmw:hasLocation ?loc).
 optional(<input> opmw:hasValue ?value).
 filter not exists{?input prov:wasGeneratedBy ?p}
}
```

The query retrieves those workflow inputs (which include both datasets and parameters) that share the same execution ?exec as datasetId and are not generated by any intermediate step ?p. The optional directive is used in this case, because parameters have a value but do not have a location, while datasets have a location but not a value.

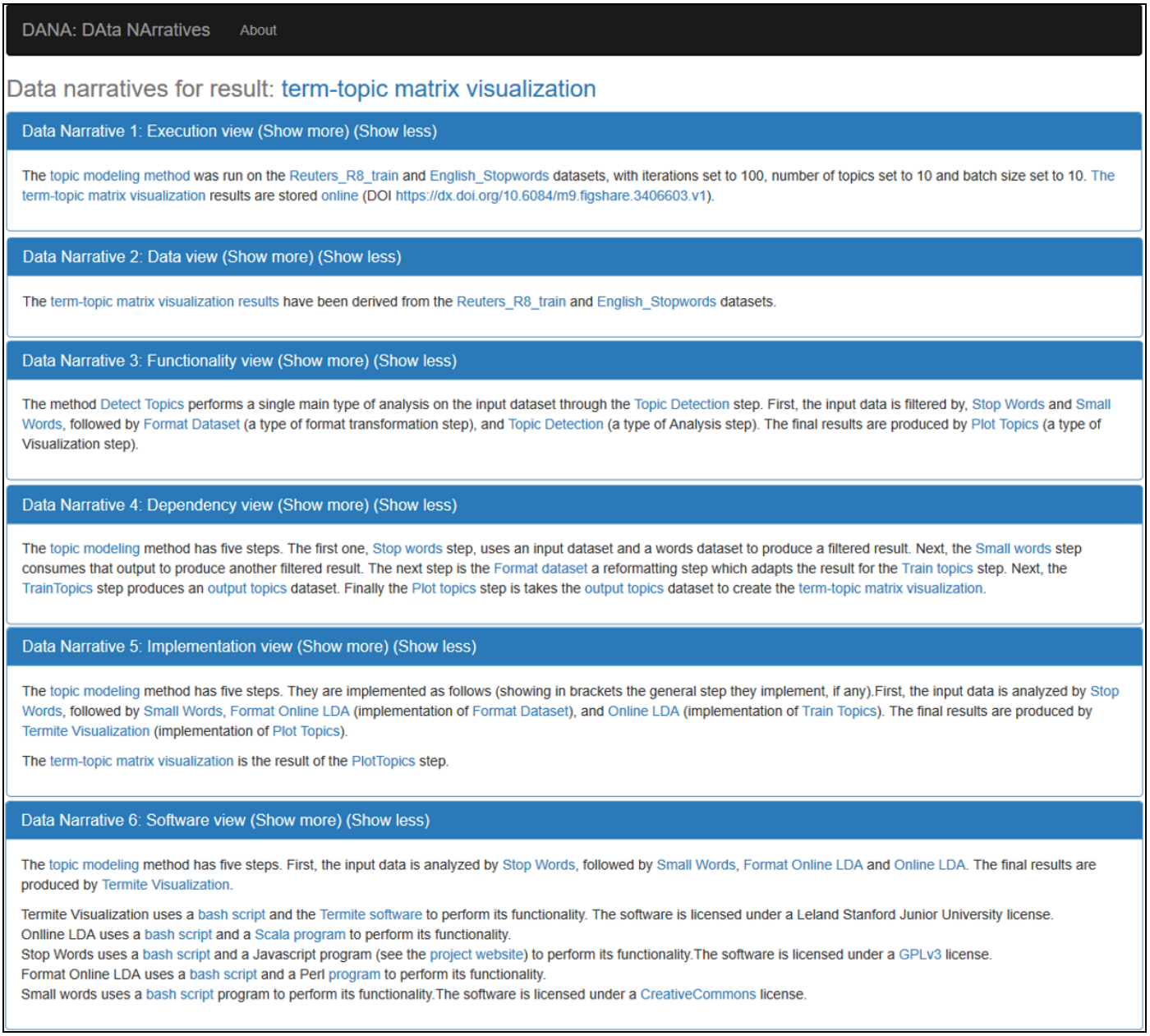

Figure 5: Data narratives automatically generated by DANA for a topic model.

Data narrative 2 requires a more elaborate query to retrieve the input datasets (?i) by navigating the provenance trace. The query is:

```
select ?i where {
 <datasetID> (prov:used/prov:wasGeneratedBy)* ?i.
 filter not exists{?i prov:wasGeneratedBy ?p}
}
```

The query retrieves all the inputs and outputs of any step that had an impact on the target dataset <datsetID> of interest. It does so in a recursive manner (represented with an asterisk in the query), by chaining the used and wasGeneratedBy properties. Since in this narrative we are only looking for the global input datasets of the workflow, we filter all those datasets that have been generated by an intermediate step ?p.

Data narrative 3 requires accessing the criticality descriptions attached to the workflow, and in particular the use of motifs. The following query retrieves the type of functionality (?f) associated to each of the workflow steps (?step), along with their inputs (?i) and outputs (?o):

```
select ?step ?f ?i ?o where {
 ?step motif:hasMotif ?m.
 ?m a ?f.
 ?step opmw:uses ?i.
 ?o opmw:isGeneratedBy ?step.
}
```

The rest of the narratives shown in Figure 5 are generated with similar explanation and queriy patterns, including the retrieval of the software metadata for data narrative 6.

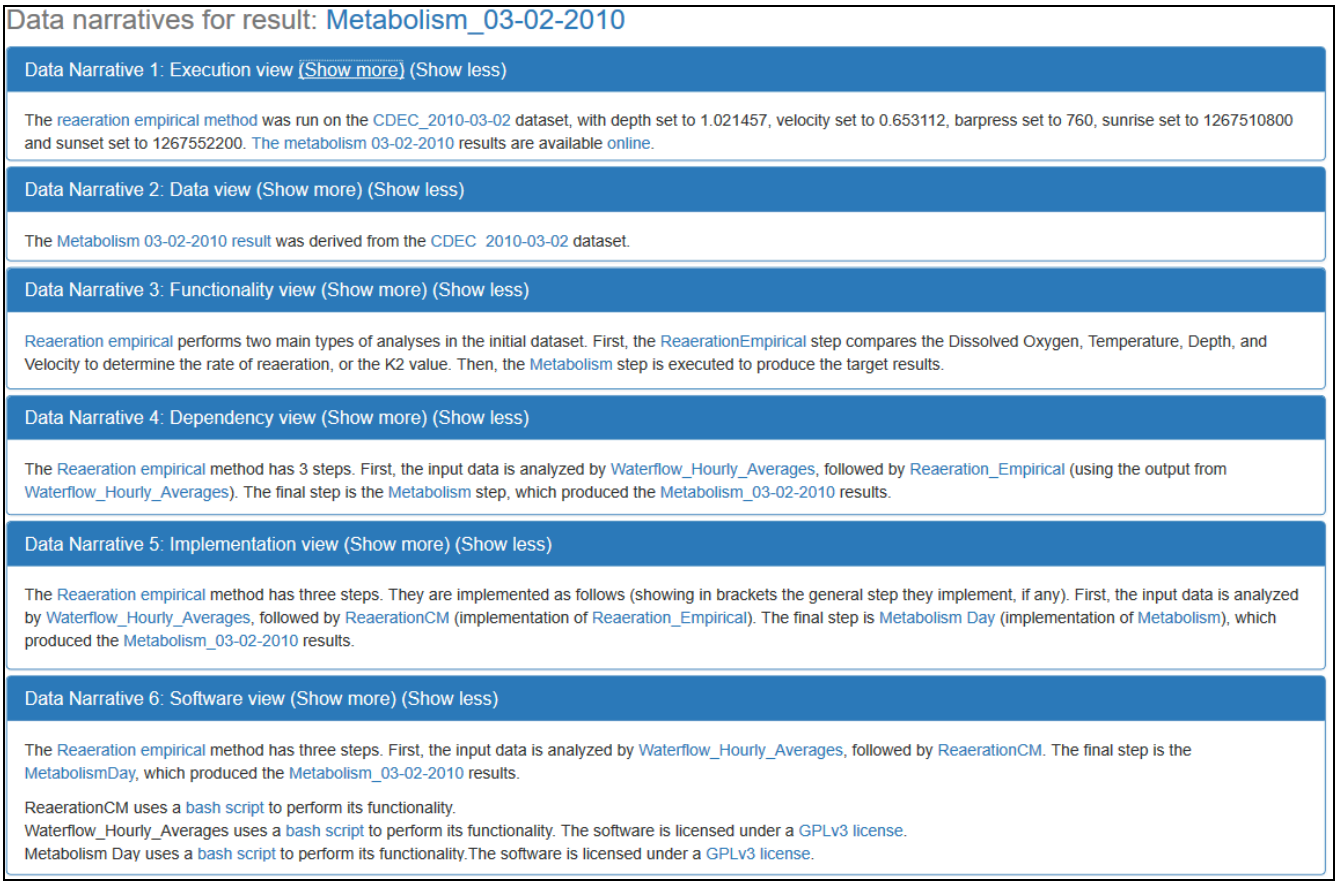

Figure 6: Data narratives generated by DANA for a water metabolism analysis.

To illustrate the generality of DANA, we show in Figure 6 the narrative accounts for a water metabolism analysis. By using the Experiment Records which include the execution, template and metadata of the workflow, the narrative accounts are created with the same explanation and query patterns as those used to generate the narrative accounts in Figure 5. We are currently applying DANA in additional scientific domains.

The software for DANA and the Experiment Records used for generating the data narratives shown in this paper is publicly available [22].

FORMATIVE EVALUATION

We did a formative evaluation of our approach with a small set of subjects. We used four workflows for web-analytics that do simple analyses, such as plotting user statistics on web accessibility, finding popular topics in a web page, detecting spam, or analyzing how users contribute to a given web page. We created a survey with six different target user scenarios, designed so that different types of narratives would be more appropriate. Each scenario consists of: 1) a description of a situation where a user has to do a task, such as a developer that needs to implement a method; 2) a workflow sketch of the analysis done; and 3) six candidate narratives of that workflow sketch. In Scenario 1 Bob is a computer programmer that is asked by his boss, Alice, to help her determine which reports were used last month to produce the visualization that appears on their website, and for this scenario the subject was asked to choose the data narrative that they think is most appropriate for Bob to use. Scenario 2 focused on generating a description of an analysis for an experienced programmer who may be familiar with the software, while Scenario 3 aimed at describing the same analysis to a novice programmer. Scenario 4 asked for a description of an analysis so as to be able to compare its results with the result from the previous year's data. Scenario 5 aimed to provide a description of an analysis for the board of directors of the company. Finally, Scenario 6 aimed at providing the details of an analysis to the system administrator of the company.

For each scenario, the subjects were asked to rate on a Likert scale how appropriate they found each narrative (1 corresponded to too little information, 5 corresponded to too much information), and then select the narrative that they considered most appropriate (if any). With these questions, we wanted to determine which narrative types were perceived as containing appropriate information and levels of detail. In addition, subjects were asked which narrative was most appropriate for that particular situation, and to provide their own narrative if they could think of a better one.

The survey was conducted online. The subjects were graduate and undergraduate students familiar with workflows but with little or no programming background. All responses were anonymous. In total, we received 12 responses. The survey [23] and the data collected [24] are available online.

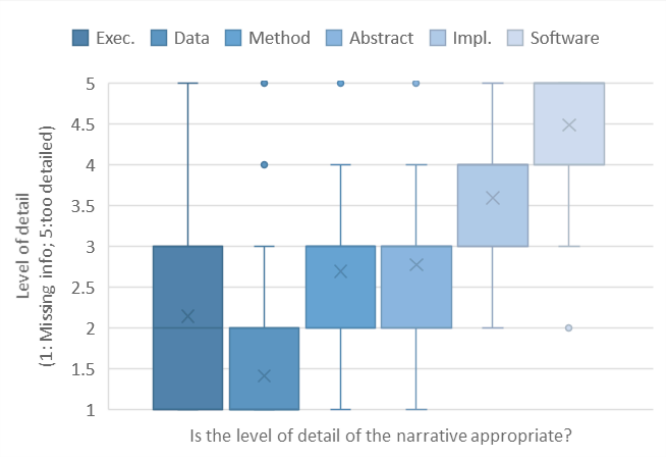

Figure 7: Level of detail of the data narratives for all scenarios.

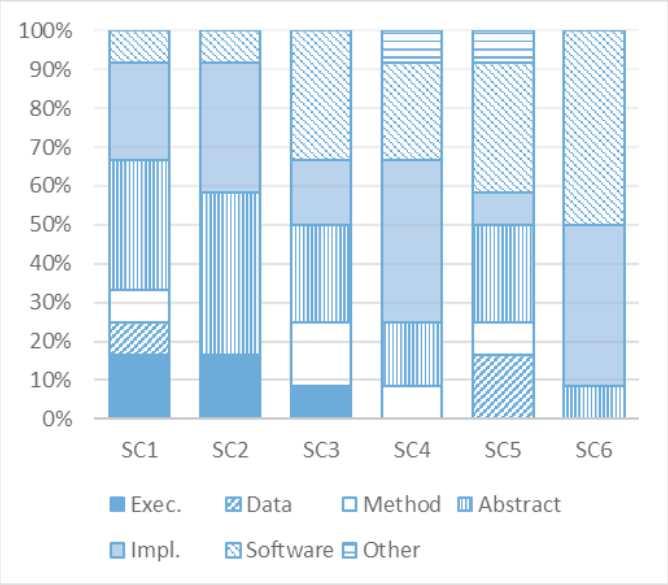

Figure 8: Distribution of the narratives best fit for addressing each scenario, according to the questionnaire results.

Figures 7 and 8 summarize the results of the survey. Figure 7 shows a box and whiskers diagram of the subjects' ratings for each type of data narrative account across all scenarios. The y axis of the figure represents the rating regarding the level of detail of the narrative account, where 1 indicates that the narrative account has missing details and 5 that the narrative account has too much information. The narrative accounts that are perceived as most appropriate are the method, abstract and implementation views, which include details about the main step functionality, the abstract description of the whole workflow and its correspondent algorithm implementation respectively. The execution and data narrative accounts, which focus solely on the data inputs and the parameters are often considered to have insufficient details for describing most scenarios. In contrast, the software narrative account which contains detailed descriptions of software is considered to be overloaded with details in most situations.

Figure 8 shows a distribution of the narratives chosen by the users as most appropriate to address each scenario, which was the final survey question for each scenario. There is no general consensus on a single type of narrative for a given scenario, except for Scenario 6 where the narrative accounts with most details (implementation and software) were rated as most appropriate. This result illustrates the need for DANA's capability for generating alternative narratives, and that users will want to choose different narratives depending on their situation.

We also included a question in the survey where we asked subjects to provide a different narrative if they would like a different one from those offered as options. There are only 2 cases where users suggested other narratives. The suggested narratives combine some aspects of the different narrative types, for example combining parameter information (execution account) with implementation details (implementation account). This result suggests that combinations of the different types of narratives would be preferred by users, and is part of our future work.

RELATED WORK

Table 1 illustrates how data narratives compare to other ways of reporting the methods of papers. Provenance records are automatically generated, true to the actual computations that took place, and enable inspection. However, they are not suitable for human consumption and are not appropriate to be included in papers. Because they focus on accurately recording the computations, they do not generally abstract from details. Visualizations of datasets present data in a human-understandable format and can be included in articles. However, they focus on presenting a dataset in itself and do not show the computational method that generated it. Also, they are often generated with some manual intervention to filter the data and to set presentation parameters. Published articles are human understandable and can have persistent identifiers (if the authors publish the data and software), and clearly abstract from details. However, because they are manually generated there is no guarantee that they are accurate with respect to the actual computational records or that they enable inspection of enough details to reproduce the work. Electronic notebooks enable inspecting and understanding the work done by the authors, but they have to be manually generated and they are not designed for different audiences. Data narratives are designed to combine useful features of provenance records, visualizations, and articles.

Narrative visualizations focus on generating an illustration that tells a story about the contents of a dataset [52]. Narrative science generates text-based stories, also for the contents of a dataset [1]. Neither uses provenance or method information, but are valuable and could become part of the generation of data narratives.

The creation of abstractions for presentation of scientific workflows is not new. ZoomUserviews* [5] explored the idea of providing different views of a scientific workflow to reduce its complexity visually. [2] proposes to annotate workflows so they can be summarized according to the functionality of their main steps. Both approaches could be combined with our work to include a visual representation of the workflow in the data narrative.

Table 1: Comparing data narratives with other approaches

Features	Data Narratives	Provenance Records	Visualizations	Articles	Electronic Notebooks
Truth to actual records	Y	Y	Just data	Maybe	Maybe
Enable inspection	Y	Y	Just data	N	Y
Human understandable	Y	N	Y	Y	Y
Abstract details	Y	N	Y	Y	N
Part of papers	Y	N	Y	Y	Maybe
Persistent	Y	Maybe	N	Y	Maybe
Different audiences	Y	N	N	N	N
Automatically generated	Y	Y	Maybe	N	N

Other related work addresses the textual summarization of provenance graphs for human consumption [48]. However, it only considers provenance information and only one text rendering, rather than using several sources of information and several text renderings as data narratives do.

There have been other efforts to publish the contents of papers in structured forms. The nanopublication model [35] describes the minimum unit of information for a fact or set of facts (e.g., datasets) contained in a paper, along with their provenance. The Research Object model [4] proposes to aggregate all the resources that are related to a scientific experiment. While these approaches capture the knowledge from a machine-readable perspective, data narrative accounts are human readable and focus on generating automatically integrated text descriptions of those interrelated resources and interlinking them within the text. The Experiment Records could be made available as a nanopublication or as a Research Object.

Finally, other related work presents frameworks with explanations for helping users debug software [40], as well as learning explanations and including feedback to personalize their results [41]. In particular, authors have started to use visualizations [45] to help explain and understand comparisons of events and text analytics data [14]. That work focuses on text analytics and other specific domains, while our work is more generally applicable to any domain with scientific workflows.

DISCUSSION

Data narratives open new avenues for research as well as practical applications. In this paper we have focused on the text generation of data narratives, but data narratives can be multimedia artifacts. Method diagrams could be easily generated to go side by side with the text and include only the steps, data, or parameters mentioned in the narrative. Tables comparing datasets and their characteristics, or software components, or entire workflows could be automatically generated. Data narratives could also include visualizations of intermediate data in addition to the final datasets and findings. The most appropriate formats for data narratives may be dependent on the field of study, the type of reader, and many other factors. Further research is needed to understand the most appropriate forms for data narratives.

Automated data narratives could be included in the methods section of papers. One approach would be to initially show default narratives (perhaps with average details in all dimensions), and let readers select narrative types and details. The system would track the most popular narratives among readers (or types of readers) and make them the new default.

There are many opportunities for publishers to exploit data narratives. Publishers could see what data narratives a reader is looking at, and based on their level of detail make recommendations for other articles to read. For example, if a reader stops at the LDA level in the explanation of the algorithm then there is no point to suggest other papers that use variants of LDA, while a reader that is looking into whether Mallet or online LDA was used may be interested in seeing a paper that uses a different LDA implementation.

Beyond enhancing how papers are written, data narratives could be a powerful extension to workflow repositories and protocol specifications (e.g., Nature protocols). Rather than relying on manually-generated documentation, or as often happens no documentation at all, automated data narratives could provide useful information for scientists to understand the descriptions and increase their reuse.

Data narratives could provide a useful paradigm for human-computer communication, particularly for intelligent systems for scientific discovery [9; 31; 27]. Data narratives could explain how a new finding was generated by the system. They could also provide the basis for new approaches to enable scientists to teach computers new scientific methods through natural language instruction (e.g., [29]).

CONCLUSIONS

In this paper we presented an approach for automatically generating data narratives to describe datasets obtained from computational methods. Data narratives can be generated from information captured in provenance records, workflows, and data and software registries. Data narratives accurately reflect the analyses performed, include persistent identifiers for all resources described, and are human readable. Data narratives can be customized for different levels of reader interest and expertise, and facilitate transparency and reproducibility. We discussed a wide range of research for future work as well as potential uses by article authors, readers, and publishers.

ACKNOWLEDGEMENTS

We gratefully acknowledge support from the Defense Advanced Research Projects Agency through the SIMPLEX program with award W911NF-15-1-0555, from the National Institutes of Health through award 1R01GM117097, and from the National Science Foundation through award ICER-1440323. We would like to thank our collaborators Parag Mallick and Victoria Stodden for their thoughtful feedback.

REFERENCES

1. Allen, N. D., Templon, J. R., McNally, P. S., Birnbaum, L., Hammond, K. J. StatsMonkey: A Data-Driven Sports Narrative Writer. AAAI Fall Symposium on Computational Models of Narrative, 2010.

2. Alper, P., Belhajjame, K., Goble, C. and Karagoz, P. Small is beautiful: Summarizing scientific workflows using semantic annotations. Proceedings of the IEEE International Congress on Big Data, June 2013.
3. Baggerly, K. A., & Coombes, K. R.. Deriving chemosensitivity from cell lines: Forensic bioinformatics and reproducible research in high-throughput biology. The Annals of Applied Statistics, 1309-1334. 2009.
4. Belhajjame, K., Zhao, J., Garijo, D., Gamble, M., Hettne, K., Palma, R., Mina, E., Corcho, O., Gomez-Perez, J., Bechhofer, S., Klyne, G., and Goble, C. Using a suite of ontologies for preserving workow-centric Research Objects. Journal of Web Semantics, 2015.
5. Biton, O., Cohen-Boulakia, S., and Davidson, S.B. Zoom*userviews: Querying relevant provenance in workflow systems. Proceedings of the Int. Conference on Very Large Data Bases, 2007.
6. Blei, D., Ng, A., and M. Jordan.“ Latent Dirichlet Allocation.” Journal of Machine Learning Research, 3: 993–1022, 2003.
7. Bourne, P. E. What Do I Want from the Publisher of the Future? PLoS Computational Biology, 2010.
8. Bourne, P. E. Beyond the PDF Workshop. Available from https://sites.google.com/site/beyondthepdf/
9. Buchanan, B. G., and D. Waltz. Automating Science. Science, 324(5923):43-44, 2009.
10. Carroll,J. J., Dickinson, I., Dollin, C., Reynolds, D., Seaborne, A. and Wilkinson, K. 2004. Jena: implementing the semantic web recommendations. In Proceedings of the 13th international World Wide Web conference on Alternate track papers & posters (WWW Alt. '04). ACM, New York, NY, USA, 74-83.
11. Chuang, J., Manning, C., and Heer, J. Termite: Visualization Techniques for Assessing Textual Topic Models. Advanced Visual Interfaces, 2012.
12. Dale, R. and E. Reiter,. Building natural language generation systems. Cambridge, U.K.: Cambridge University Press, 2000.
13. De Roure, D., Goble, C., Stevens, R. The design and realizations of the myExperiment Virtual Research Environment for social sharing of workflows. Future Generation Computer Systems, 25 (561-567), 2009.
14. Dinakar, K., Chen, J., Lieberman, H., Picard, R., and R. Filbin. Mixed-initiative real-time topic modeling & visualization for crisis counseling. Proceedings of IUI 2015, 417–426, 2015.
15. Bracewell, D. J., Fredericksen, C., and M. Dillinger. Students’ Strategies for Writing Instructions. Written Communication, 9(2), 1992.
16. Donoho D.L., Maleki, A., Rahman, I.U., Shahram, M., and V. Stodden. Reproducible Research in Computational Harmonic Analysis. Computing in Science & Engineering 11(1), 2009.
17. Garijo, D. Mining abstractions in scientific workflows. PhD thesis. Escuela Tecnica Superior de Ingenieros Informáticos, Universidad Politecnica de Madrid. 2015.
18. Garijo, D. and Y. Gil. A New Approach for Publishing Workflows: Abstractions, Standards, and Linked Data. Proceedings of the Sixth Workshop on Workflows in Support of Large-Scale Science (WORKS'11), held in conjunction with the IEEE ACM International Conference on High-Performance Computing (SC), 2011.
19. Garijo, D. and Gil, Y. Augmenting PROV with plans in P-Plan: Scientific processes as Linked Data. In Second International Workshop on Linked Science: Tackling Big Data (LISC), held in conjunction with the International Semantic Web Conference (ISWC), Boston, MA. 2012.
20. Garijo, D., Kinnings, S., Xie, L., Xie, L., Zhang, Y., Bourne, P.E., and Y. Gil. Quantifying Reproducibility in Computational Biology: The Case of the Tuberculosis Drugome. PLOS ONE, 2013.
21. Garijo, D., Alper, P., Belhajjame, K., Corcho, O., Gil, Y., and C. Goble. Common Motifs in Scientific Workflows: An Empirical Analysis. Future Generation Computer Systems, 36, 2014.
22. Garijo, D., and Gil, Y. dgarijo/DataNarratives: Pre-release: Data Narratives [Data set]. Zenodo. http://doi.org/10.5281/zenodo.232075. 2017.
23. Garijo, D., and Gil, Y. Questionnaire for evaluating data narratives [Data set]. Zenodo. http://doi.org/10.5281/zenodo.234648. 2017.
24. Garijo, D., and Gil, Y. Evaluation results for Data Narratives [Data set]. Zenodo. http://doi.org/10.5281/zenodo.234660. 2017.
25. Garijo, D., Gil, Y., and Corcho, O. Abstract, Link, Publish, Exploit: An End to End Framework for Workflow Sharing. To appear in Future Generation Computer Systems, 2017.
26. Gil, Y. Human Tutorial Instruction in the Raw. ACM Transactions on Interactive Intelligent Systems, 5(1), 2015.
27. Gil, Y. and Pierce, S. A. (Eds). Final Report of the 2015 NSF Workshop on Information and Intelligent Systems for Geosciences. National Science Foundation Workshop Report, October 2015.
28. Gil, Y. Ratnakar, V., Kim, J., Gonzalez-Calero, P. A., Groth, P., Moody, J. and E. Deelman. Wings: Intelligent Workflow-Based Design of Computational Experiments. IEEE Intelligent Systems, 26(1):62-72, 2011.
29. Gil, Y., Ratnakar, V., and Fritz, C. TellMe: Learning Procedures from Tutorial Instruction. In Proceedings of the ACM International Conference on Intelligent User Interfaces, Palo Alto, CA, 2011.
30. Gil, Y., Ratnakar, V., Verma, R., Hart, A., Ramirez, P., Mattmann, C., Sumarlidason, A., and S. L. Park. Time-Bound Analytic Tasks on Large Datasets through

Dynamic Configuration of Workflows. Proceedings of the Eighth Workshop on Workflows in Support of Large-Scale Science (WORKS), 2013.

31. Gil, Y., Greaves, M., Hendler, J., and H. Hirsh. Amplify scientific discovery with artificial intelligence. Science, 346(6206), 2014.
32. Gil,Y., Ratnakar, V., and Garijo, D. OntoSoft: Capturing Scientific Software Metadata. Proceedings of the Eighth ACM International Conference on Knowledge Capture, Palisades, NY, 2015.
33. Gil, Y., David, C. H., Demir, I., Essawy, B. T., Fulweiler, R. W., Goodall, J. L., Karlstrom, L., Lee, H., Mills, H. J., Oh, J., Pierce, S. A, Pope, A., Tzeng, M. W., Villamizar, S. R., and Yu, X. Towards the Geoscience Paper of the Future: Best Practices for Documenting and Sharing Research from Data to Software to Provenance. Earth and Space Science, 3, 2016.
34. Groth, P., and Gil, Y. Analyzing the Gap between Workflows and their Natural Language Descriptions. In Proceedings of the IEEE International Workshop on Scientific Workflows (SWF), 2009.
35. Groth, P., Gibson, A., Velterop, J. The anatomy of a nanopublication. Information Services and Use, 30, 1-2: 52-56, 2010.
36. Hoffman, M., Blei, D., and F. Bach. Online Learning for Latent Dirichlet Allocation. NIPS, 2010.
37. Ince, D. C., Hatton, L. and J. Graham-Cumming. The Case for Open Computer Programs. Nature, Vol 482, 2012.
38. Kinnings, S. L., Xie, L., Fung, K. H., Jackson, R. M., Xie, L, et al. The Mycobacterium tuberculosis Drugome and Its Polypharmacological Implications. PLoS Comput Biol 6(11), 2010.
39. Kluyver, T., Ragan-Kelley, B., Pérez, F., Granger, B., Bussonnier, M., Frederic, J., et al. Jupyter Notebooks—a publishing format for reproducible computational workflows. Positioning and Power in Academic Publishing: Players, Agents and Agendas, 2016.
40. Ko, A. J., and Myers, B. A. Debugging reinvented. Proceedings of the 30th international conference on Software engineering, 301–310, 2008.
41. Kulesza, T., Burnett, M., Wong,W.-K., and S. Stumpf. Principles of explanatory debugging to personalize interactive machine learning. Proceedings of IUI 2015, 126–137, 2015.
42. Lebo, T., McGuiness, D., Belhajjame, K., Cheney, J., Corsar, D., Garijo, D., Soiland-Reyes, S., Zednik, S., and J. Zhao. The PROV ontology, W3C recommendation. Technical report, WWW Consortium, 30th April 2013.
43. Manola; F., Miller, E. *RDF Primer*. W3C Recommendation. WWW Consortium. 10 February 2004.
44. Mates, P., Santos, E., Freire, J., and Silva, C. T. CrowdLabs: Social analysis and visualization for the sciences. 23rd International Conference on Scientific and Statistical Database Management (SSDBM), 2011.
45. Malik, S., Du, F., Monroe, M., Onukwugha, E., Plaisant, C., and B. Shneiderman. Cohort comparison of event sequences with balanced integration of visual analytics and statistics. In Proceedings of IUI 2015, 38–49, 2015.
46. McCallum, A. K. MALLET: A Machine Learning for Language Toolkit. http://mallet.cs.umass.edu. 2002.
47. Mesirov, J. P. Accessible Reproducible Research. Science, 327:415, 2010.
48. Moreau, L. Aggregation by provenance types: A technique for summarising provenance graphs. arXiv preprint arXiv:1504.02616, 2015.
49. Reality Check on Reproducibility. Nature 533(7604), 2016.
50. ReadCube. https://www.readcube.com/.
51. Prud'hommeaux, E. and Seaborne, A. (Eds). "SPARQL Query Language for RDF". W3C Recommendation, 15 January 2008. https://www.w3.org/TR/rdf-sparql-query/
52. Segel, E. and Heer, J. Narrative Visualization: Telling Stories with Data. IEEE Trans. Visualization & Comp. Graphics (Proc. InfoVis), 2010.
53. Steehouder, M., Karreman, J. and Ummelen, N. Making sense of step-by-step procedures. Proceedings of 2000 Joint IEEE International and 18th Annual Conference on Computer Documentation (IPCC/SIGDOC 2000).
54. Stodden, V., McNutt, M., Bailey, D. H., Deelman, E., Gil, Y., Hanson, B., Heroux, M. A., Ioannidis, J. P., and Taufer, M. Enhancing Reproducibility for Computational Methods. Science, 354. 2016.
55. Van Noorden, R. Sluggish data sharing hampers reproducibility effort. Nature, 2015.
56. Wang, Y., Bai, H., Stanton, M., Chen, W., and E. Y. Chang. PLDA: Parallel Latent Dirichlet Allocation for Large-Scale Applications. AAIM, 2009.
57. Wolfram Research, Inc., Mathematica, Version 10.4, Champaign, IL, 2016.
58. Xie Y. knitr: A Comprehensive Tool for Reproducible Research in R. In Victoria Stodden, Friedrich Leisch and Roger D. Peng, editors, Implementing Reproducible Computational Research. Chapman and Hall/CRC. 2014. ISBN 978-1466561595.