WHITE PAPER

A Novel Scientific Data Repository Framework

The Scientific Data Repository Framework (SDRF) encompasses data models, publishing guidelines, and code libraries to help authors and publishers deploy open-access research data sets that are associated with scientific publications. Nowadays there are many general-purpose and domain-specific portals hosting scientific data; there are also several available formats for describing and encoding scientific data, such as Research Objects, schema.org/Dataset, Digital Curation Center, SciData, BioCoder, and MIBBI (Minimum Information for Biological and Biomedical Investigations). However, SDRF is unique in merging different data-set formats into a unified, overarching standard which can be adapted to different publishing environments.

In order to conform to current specifications — such as FAIRsharing (Findable, Accessible, Interoperable, Reusable) or the Bill and Melinda Gates Foundation guidelines for authors — proper protocols must be implemented at several stages of the publishing process. In particular, (1) publications themselves should provide clear descriptions of accompanying data sets and where that data is hosted; (2) publication repositories (such as Springer Nature) should make data-set links and metadata clearly visible on web pages where documents are read or previewed; (3) data sets themselves need to include metadata and supporting files which help researchers properly access, visualize, and reuse the data; and (4) data sets need to be connected with software which has the correct features to load and display the relevant raw data files. SDRF will include technology applicable to each of these four facets of the publishing and data-sharing pipeline.

It is important to emphasize that data sets are only truly valuable if they are machine-readable and seamlessly integrated into domain-specific software ecosystems. Scientists who carefully examine and reuse published data sets are usually researchers doing technical work in a field closely related to the original authors'; in many cases there are specialized software applications, computational methods, algorithms, and research protocols which are endemic to the relevant subject areas. When sharing research data, accordingly, publishers and authors should make it as easy as possible for scientists to examine the data within the digital ecosystem they utilize for their own research. This often means that document viewers — e.g., PDF viewers and/or HTML pages on publisher portals — should ideally be interconnected with scientific applications, so that scientists, when reading books/articles, can seamlessly launch domain-specific software and visualize/examine associated data sets. Unfortunately, most scientific software does not incorporate code libraries to parse metadata describing open-access data sets. Therefore, publisher and data-hosting repositories can only be truly interoperative with scientific applications by providing plugins or extensions that add data-set-accession capabilities to existing scientific software.

To demonstrate how such plugins can work, as well as other facets of the data-publication process augmented via SDRF, this paper will review two case-studies involving existing or forthcoming articles.

First Case Study: "Parkinson's Disease Diagnosis: Effect of Autoencoders to Extract Features from Vocal Characteristics" from the International Journal of Speech Technology, forthcoming

This case study demonstrates a scenario where one article reuses multiple pre-existing data sets. The article, by Ashena Gorgan Mohammadi, Pouya Mehralian, Amir Naseri, and Hedieh Sajedi, examines Parkinson's Disease symptomology from multiple perspectives, including gait (loss of motor function), speech impairment, and bioimaging (MRIs). The authors apply Machine Learning to data sets focused on these three different diagnostic areas, advansing research to refine Parkinson's predictors and diagnoses. The overall information can be summarized as follows:

1. Gait data was primarily drawn from a PhysioNet data set (https://physionet.org/content/gaitpdb/1.0.0/) obtained via sensors attached to subject's feet as they walked (the study includes both Parkinson's patients and healthy controls). This sensor data is provided as a collection of text files, each file corresponding to one patient (or control subject), with each line in a file representing a single time snapshot. The lines are divided into space-separated columns each representing force exerted on a single sensor (plus two columns for total force on the left-foot and right-foot sensors respectively). This data set also includes demographic and clinical information for each patient, in a spreadsheet format.

   An additional source of gait data used by the authos is a more recent (2019) study whose data is shared only on request (see https://www.nature.com/articles/s41598-019-53656-7#MOESM1).
2. The primary source of speech data is a data set hosted by the University of California Irvine Machine Learning Repository (https://archive.ics.uci.edu/ml/datasets/Parkinsons). The central information is a **CSV** file, where each line represents a single voice recording from one of 23 Parkinson's patients or 8 healthy controls (each subject made multiple recordings). The individual lines present a quantitative model of subjects' speech via a collection of acoustic features/attributes.

Nowadays many authors publish data sets to go along with books/chapters/articles, and it is important for document portals (like Springer Nature) to properly identify these data sets when they are available. The basic concept is simple: each data set has its own digital identifier, akin to (but distinct from any) Document Object Identifier pointing to a publication. The ids for data sets and their concomitant publications can be linked on publishers' web sites (and on external resources such as CrossRef).

But while the goal is simple, there are many details that have to be worked out, and publishers thus far have failed to implement data-set references in ways that conform in spirit or in practice to recommendations stipulated by groups like FAIRsharing or the Bill and Melinda Gates Foundation (who has detailed data-sharing guidelines applicable to researchers receiving BMGF funding). Some of the issues are as follows:

1. There are multiple formats proposed for describing research data. Standards such as Research Object Bundles, schema.org/Dataset, and Digital Curation Center specify how data sets should be documented – that is to say, metadata such as authors, versions, dates, digital ids, data formats, provenance, and other details about raw data (this is separate and apart from raw data itself). Other standards define how the data should be scientifically documented or structured, which is less a matter of publication info (like authorship), so much as concerned with data sets' scientific properties and/or research methods: SciData, MIBBI (Minimum Information for Biological and Biomedical Investigations), SciXML, BioCoder, etc. Ideally, publishers should use a common data model which is compatible with these different formats.
2. Although there are common standards for representing research data and publication text, every publisher and data-hosting platform has their own technology. As such, precise protocols for interoperating with different platforms have to be specified individually. One way to achieve this is to construct a "Scientific Data Repository Model" (SDRM) tailored to each distinct publication and/or data-set corpus/repository. Each SDRM can then form the basis of an "SDRM Module" which is a code library for interoperating with portals modeled via the corresponding SDRM.
3. Publications often have graphics or multimedia content that require special viewers. More often than not these assets are visual or structured representation of data which may be placed in a data set – e.g. a table printed inline in the document is derived from a spreadsheet which is its own data file. Many document portals allow readers to examine these inline figures in separate frames and windows. To support more sophisticated multimedia content, with file types specific to different scientific fields, publishers are attempting to create browser plugins to visualize complex scientific data. In general, however, such scientific assets should really be viewed in domain-specific scientific software. This suggests that publishers' portals and document viewers should interoperate with existing applications – rather than, or at least in

addition to, trying to emulate these applications via browser plugins. One facet of a publishers' SDRM would be how their portals identify and communicate with the "preferred software" that would be used to view different data sets.

4. Data Sets have multiple dimensions, not just raw data. This is particularly true in light of protocols such as FAIR (Findable, Accessible, Interoperable, Reusable) – raw data by itself is not especially interoperable or reusable. According to FAIR, authors should supplement raw data files with information or computer code to help researchers use/analyze/visualize their data. This may include providing "workspace" files which configure applications when data sets are loaded into the preferred software.
5. Aside from overall links between data sets and publications, parts of a publication may be linked to parts of a data set. For example, a table, scatter-plot, or plot-graph displayed in a publication may be linked to a .csv file in a data set. Correlations between publications and data sets may also be based on Controlled Vocabularies and other ways of identifying concepts in publication texts. If publications and data sets are both annotated using the same Controlled Vocabulary, then links are implicitly created between the two resources, wherever a part of a data set and of a publication text, respectively, are annotated as concerning the same concept. To make these connections explicit, publication manuscripts should ideally be annotated with concepts or data structures that directly or indirectly define granular connections between publications and data sets. When publications are fully open-access, these annotations can be included in machine-readable full-text resources that can be freely downloaded.

Connections between publications and data sets can be asserted by using a micro-citation system within the data set and annotations within the publication. Here are a few examples: the name of a protein may be marked in publication text and also linked to a Protein Databank file in a data set; the name of a flourochrome may be marked in a publication and cited as one channel group in a Flow Cytometry file; the description of an Image Segmentation algorithm, such as a Sobel-Feldman operator, may be linked to an image in a data set resulting from that operator applied to a primary image, and also mentioned in the text as part of the description of an image-processing workflow; biochemical assays may be described in the text and also formally declared via MIBBI, BioCoder, or SpringerProtocols, with that systematic info included in the data set; and so forth.

There are then, in general, four or five different dimension to a data set – or at least any data set which is curated according to current standards. In addition to (1) raw data, there is (2) metadata describing the data set (file formats and sizes, authorship, provenance, etc.); (3) formal declarations of research/lab methods or protocols; (4) workspace files for integrating with software that displays the raw data; and (5) annotated machine-readable full-text. Of these, (3), (4) and/or (5) may or may not be present, but all four or five will be part of thorough, well-curated data sets (excepting lab-methods descriptions, which are only applicable to lab-based research – although similar formats can be used to document computational workflows as well). The metadata helps researchers determine how to view and use the raw data. For example, it helps them select which application to use to open raw data files. Once they are using the proper application, workspace files configure the application for optimal interaction with that particular data set. Examples of workspace files include ParaView State Files, FlowJo Workspaces, MeshLab Project Files, CaPTk extensions, Jupyter or Kaggle notebooks, etc.

When publishing data sets, authors need to combine raw data files with these other kinds of files (metadata, workspace, full-text) into "packages" or "bundles;" we prefer to use the term "Supplemental Archive." These unified resources are then given distinct identifiers and linked to publications. Several tools exist to help authors with this step, such as Elsevier's Research Object Composer, and DataPackageR (funded by the Bill and Melinda Gates Foundation). But these tools (or others like them) are poorly integrated into publishing platforms. Ideally, publishers should do at least the following:

1. When authors submit manuscripts with associated Supplemental Archives, publishers should insert publication-specific information (such as publication dates, web links, full-text files for

open-access documents, and metadata such as info about peer-review process as appropriate) into the archive. Manuscripts should be checked to make sure they properly reference the Supplemental Archive (e.g. via a footnote or a digital identifier in the main text). When a publication discusses data sets linked to other publications, manuscripts should be checked to make sure they cite those data sets directly – they should not only cite the publications which introduce those data sets.

2. When publications with associated Supplemental Archives are published on portals (such as Springer Nature), hyperref links to the Supplemental Archives should be clearly visible and accessible – without the reader having to skim article text or scroll down to a "Supplemental Materials" or "Data Availability" statement.
3. Portals (such as Springer Nature) should read pertinent metadata from the Supplemental Archive and make it clearly visible to readers to help them decide whether they wish to download a data set. Important information includes: how large is the data set overall; what file types are included in the data set; what are common applications used to view these files; what is the digital id for the data set; and links to the portals where the data sets are hosted.
4. Publishers should also consider using collapsible iFrames or secondary windows to show a data set preview (a list of raw data files) and information about file types and applications. This material could include links to web sites where readers can download software needed to view the raw data files if they do not have such software already.
5. Data set info displayed visually for readers on document portals should also be incorporated into APIs. At the minimum, an API endpoint should yield data set ids given document ids. A more complete set of API queries for data sets might include:

   (a) Search for publications based on attributes of their associated data sets (e.g.: file formats/programming languages; recommended data-viewing software; keywords in data-set descriptions; protocols as identified by standardization formats like MIBBI; repository host where the data is stored);
   (b) Return a complete data-set description object serialized for use by an API client library, given a data-set identifier;
   (c) Individual API endpoints for the most crucial metadata given a data-set identifier (e.g.: file format; size; authorship; recommended applications);
   (d) Handle requests for microcitations generated by document signals, the same signals used for inter-application networking between document viewers and scientific applications;
   (e) Provide API endpoints for finding and using plugins (discussed below).

6. Publishers should register document-to-Supplemental-Archive links on third-party bibliographic services, like CrossRef and Semantic Scholar.
7. Ideally, publishers should provide plugins or standalone applications for readers to use after they have downloaded data sets. While readers can open raw data files in the proper applications, those applications may not have the capabilities to understand the other files in a Supplemental Archive. Publishers should therefore provide tools or plugins so that these applications can access the whole archive, not just the raw data. If portals link to web sites where readers can download the applications for viewing raw data files, these links could be paired with instructions on how to obtain and use plugins provided by the publisher.

Fully integrating data sets, publications, and data-visualization/management software often requires a triangular relationship between different software components, which have distinct roles. For a concrete example, consider a prototype which we are developing for Flow Cytometry data: raw data in this context is stored in Flow Cytometetry Standard (.fcs) files, which need to be parsed with special code libraries. Our prototype uses cytoLib, an open-source C++ implementation, for interoperating with FCS files. The parsed data then needs to be sent to a component designed for visualizing FCS experimental results. Normally, cytoLib interoperates with libraries coded in R (the statistical programming language) for data visualization; however, our prototype is based on FACSanadu, a Java library we are porting to C++. So the application logic is split between a data

accession/parsing library (cytoLib) and a GUI/visualization library (FACSanadu), both of which need to be modified to work in an SDRM context. Finally, a third component is needed to properly connect the data-parsing and data-visualization capabilities, and to extract the information required by both of these components from Supplemental Archives. This kind of triangular relationship – involving a data parser, data viewer, and archive manager – is likely to be a recurring pattern for implementing SDRM modules. In general, individual SDRM modules would provide computer code instantiating such "SDRM Triangles" for different scientific disciplines.

To demonstrate how plugins and/or Supplemental Archives can work in general, consider the connections which exist between parts of publications (e.g., a figure illustration) and corresponding parts of data sets (e.g., .csv files). Here are some illustrative use-cases:

1. A graphic in a publication displays a segmented image. Via a context menu, this graphic can be linked to an image-processing data set which is viewed in image-annotation software (e.g. IAT, from the University of Milano-Bicocca) showing segments as annotations; or the data set is viewed in image-analysis software (like CaPTk, from the University of Pennsylvania) where researchers can reproduce the steps taken to obtain the segmented image from the original.
2. A figure shows one rendering of a 3d manifold showing the distribution of a high-dimensional data set under dimensional reduction. Via a context menu, this graphic can be viewed in 3d with (for instance) ParaView.
3. A graphic shows Flow Cytometry data (generated by probing cells and cellular-scale organic materials with lasers and fluorochromes) as part of a serological assay. Via a context menu, the "event data" yielding that graphic – together with gating information, which is similar to image annotation – can be viewed in Flow Cytometry software, such as FlowJo or FACSanadu. The event data itself is stored in FCS files (which may be accessed from web resources, such as flowRepository) while gating information is provided by FlowJo workspace files or GatingML-style files internal to the data set.
4. A graphic shows a 2d or 3d outline of the molecular structure of a protein. Via a context menu, this structure can be viewed as an interactive 3d presentation in IQmol, where the raw data is obtained from Protein Data Bank. As a variation on this use-case, every protein mentioned in a chemistry paper can be marked with a hyperlink and/or context menu launching IQmol with the relevant .pdb file.
5. An archaeology paper shows images from a site which has been investigated with advanced imaging/laser equipment, e.g. point-cloud scanners or Matterport 360-degree cameras. Via context menus or links embedded in figure captions, the associated data set may be viewed in 3d graphics software such as MeshLab, or in Panoramic Photography viewers such as Panini.
6. A computer science paper includes code samples; context menus or hyperlinks at the start of each sample can be designed to open an IDE (e.g. Qt Creator) or an interactive programming notebook (e.g., Jupyter) to run the sampled code, with the complete code provided by a data set.
7. A paper presenting lab results includes a schematic summary of lab protocols generated by BioCoder. A context menu can then launch Qt Creator to display the actual BioCoder files, included in a data set.
8. A paper discussing a vaccination campaign references data sets evaluating subjects' immunological responses, geospatial visualization of the campaign's target area, and epidemiological modeling. This paper may then be linked to several different scientific applications: ArcGIS for the geospatial data; FlowJo for the immunology; Qt Creator for running epidemiological simulations (modeled via the EpiFire library, for instance).
9. A paper on stroke rehabilitation may use DICOM to record both image data (e.g. MRIs) and audio data (e.g. documenting aphasia). These DICOM files need to be opened in the proper software – e.g. medInria for 2d images, ITK Snap for 3d images, and Audacity for audio. When the publication text describes the diagnostic significance of particular DICOM-encoded findings, then (perhaps via DICOM Structured Recording), these text segments should be annotated to

identify the software category appropriate for the specific DICOM files used.

For each of these scenarios, there must be a functioning integration between the document-viewers which readers use for the main publication, and the domain-specific applications they use for the data set files. One way to achieve this is for publishers to provide a suite of plugins to scientific/technical applications used to view most kinds of data files that are published in conjunction with scientific literature. Each plugin would be configured to recognize signals generated from documents created via that publisher's software and to respond to those signals properly (retrieving and opening data files from the relevant open-access data sets).