Proposing a CORD-19 Software Development Kit to Improve Machine Readability and Text Mining

LTS is founded by Amy Neustein, PhD, Series Editor of **Speech Technology and Text Mining in Medicine and Health Care** *(de Gruyter); and Editor of* **Advances in Ubiquitous Computing: Cyber-Physical Systems, Smart Cities, and Ecological Monitoring** *(Elsevier, forthcoming). These publishers have placed Dr. Neustein's publications on their open access portals linked from* CORD-19.

CORD-19 (the "Covid-19 Open Research Dataset") is a new coronavirus data collection which was released in conjunction with a White House initiative to spur Covid-19 research. This initiative is described as a "call to action ... to develop new text and data mining techniques that can help the science community answer high-priority scientific questions related to Covid-19" (see https://www.whitehouse.gov/briefings-statements/call-action-tech-community-new-machine-readable-covid-19-dataset/). The White House is spearheading a consortium of industry and academic institutions, led by the Allen Institute for AI Research, which curated a "machine-readable Coronavirus literature collection" that includes article metadata and (in most cases) publication text, both encoded as JSON files, for over 44,000 coronavirus research papers. This corpus of publication texts and metadata is also paired with links to publisher portals (including Springer Nature, Wiley, Elsevier, the American Society for Microbiology, and the New England Journal of Medicine), so as to provide scientists with full open access to selected Covid-19-related literature; these resources collectively constitute CORD-19 (see [2]).

Linguistic Technology Systems (LTS) would like to create a Software Development Kit (SDK) to help scientists utilize CORD-19. The foundation of the SDK is a framework that we call "Annotation Exchange Format" (AXF), which is designed to facilitate document preparation as well as interoperability between text and/or data mining tools. AXF is employed as a querying format that is integrated with data sets which will accompany the forthcoming volume, *Advances in Ubiquitous Computing*, which is part of Elsevier's Series, *Advances in Ubiquitous Sensing Applications for Healthcare*. The AXF framework includes an SDK oriented to the construction of document archives organized around particular themes or topics; our proposed SDK for CORD-19 would therefore be a specialization and extension of our generic AXF SDK. Our proposed CORD-19 SDK would include new code libraries explicitly implemented for data-management operations specific to CORD-19. It would also include a package of applications, modified to support Covid-19 research, that would collectively create an integrated and self-contained computing environment. These three parts of the SDK — the new code libraries, the application package, and AXF — are outlined in this paper.

I New Code Libraries within the Proposed SDK

The CORD-19 collection was formulated with the explicit goal of promoting both *text mining* and *data mining* solutions to advance coronavirus research. This means that CORD-19 is intended to be used both as a document archive for text mining and as a repository for finding and obtaining coronavirus data for subsequent research. Because the White House announcement requests institutions to develop additional technologies which would help scientists and jurisdictions to take advantage of CORD-19, the collection was released with the anticipation that industry and academia would augment the underlying data by layering on additional software. Our proposed CORD-19 SDK would do just that: this SDK would serve as a component that would provide analytic capabilities to make the raw CORD-19 data more valuable; it would also serve as a toolkit through which other developers could create new solutions targeting the CORD-19 repository.

To accomplish these goals, our proposed SDK would include a collection of new code libraries to aid programmers in the implementation of algorithms to investigate the CORD-19 corpus. These code libraries would enhance the underlying data by providing the following useful features:

Datatypes Encapsulating Access to the Semantic Scholar API This library would provide a Qt-based wrapper around Semantic Scholar API calls, and model certain objects relevant to this API, such as entities and their mentions, citations, and authorship links. One use-case for such a library concerns document preparation, where the API may be valuable for obtaining information such as Semantic Scholar paper ids and entity strings, which can be employed when building metadata files embedded in publications and/or to add features to human-readable manuscripts (examples in the second case include providing Semantic Scholar paper ids within bibliographic entries and selecting keywords based on entities mentioned within cited papers).

Tools for Correcting Transription Errors Transription errors can cause the machine-readable text archive to misrepresent the structure and content of documents. For instance, there are cases in CORD-19 of scientific notation and terminology being improperly encoded. As a concrete example, "**2'-C-ethynyl**" is encoded incorrectly in one CORD-19 file as "**2 0 -C-ethynyl**" (see [3] for the human-readable publication where this error is observed; the corresponding index in the corpus is 9555f44156bc5f2c6ac191dda2fb651501a7bd7b.json). To help address these sorts of errors — which could stymie text searches against the CORD-19 corpus — our SDK would augment the CORD-19 repository by providing alternate machine-readable encodings of the archived documents in formats such as XML, whenever they are available, as a supplement to CORD-19's JSON representation. The SDK would then provide tools to cross-reference multiple versions of each document, so as to correct errors in the original JSON encodings.

Tools for Converting Between Data Formats Although the CORD-19 corpus is published as JSON files, many text-mining tools such as those reviewed in [7] recognize input in alternative formats, such as XML, BioC, or JSON trees with different schema than CORD-19. Our proposed SDK would provide libraries to read CORD-19's JSON files and output data in one of these alternative formats, so as to initiate a text mining workflow. The SDK would also include tools for manipulating the *results* of text mining algorithms, which is often represented in formats such as XML and CoNLL (Conference on Natural Language Learning; this is a schema for representing sentences via parse-graphs).

An LAF Architecture for Annotations over CORD-19 The structure of CORD-19 allows text segments to be defined via a combination of JSON file names, paragraph ids, and character indices. Consequently, the archival schema's mechanism for standoff annotation of intra-document references may be extended for assertions of linguistic, resource, or topical annotations from multiple Annotation Spaces. The proposed SDK, in particular, would provide code for additional standoff annotation categories of the kinds commonly used in biomedical text mining. As a concrete example, the corrected text segment "**2'-C-ethynyl**" mentioned earlier can be annotated as a molecular component.

Tools for Research Data-Mining Even though many papers in CORD-19 are paired with published data sets, there is currently no tool for locating research *data* through CORD-19. For example, the collection of manuscripts available through the Springer Nature portal linked from CORD-19 includes over 30 Covid-19 data sets, but researchers can only discover that these data sets exist by looking for a "supplemental materials" or a "data availability" addendum near the end of each article. These Springer Nature data sets encompass a wide array of file types and formats, including FASTA (which stands for Fast-All, a genomics format), SRA (Sequence Read Archive, for DNA sequencing), PDB (Protein Data Bank, representing the 3D geometry of protein molecules), MAP (Electron Microscopy Map), EPS (Embedded Postscript), CSV (comma-separated values), and tables represented in Microsoft Word and Excel formats. To promote data mining in the context of CORD-19, our SDK would (1) maintain an index of data sets linked to CORD-19 articles and (2) merge these resources into a common representation (such as XML) wherever possible.

Wrappers for Network Requests Scientific use of CORD-19 will often require communicating with remote servers. For example, genomics information in the Covid-19 data sets (such as those mentioned above that are available through Springer Nature) is generally provided in the form of accession numbers which are used to query online genomics services. Similarly, text mining algorithms often rely on dedicated servers to perform Natural Language Processing; these services might take requests in BioC format and respond with CoNLL data. As another case study epidemiological studies of Covid-19 may need to access APIs or data sets such as the John Hopkins University "dashboard" (see https://coronavirus.jhu.edu/map.html, which is paired with a GIT archive updated almost daily). To reduce the amount of "biolerplate code" which developers need for these networking requirements, our company's SDK would provide code libraries based on the Qt Networking Module to manage networking requests and responses. Programmers would therefore have a unified framework with which to construct remote queries and route responses, a framework which could be used across disparate scientific disciplines (genomics, NLP, epidemiology, and so forth).

In short, the code libraries decribed above would augment the value of CORD-19 by providing tools out-of-the-box to help scientists (and their codewriters) leverage CORD-19 data. Although we can expect that numerous code libraries will be implemented so that researchers can use CORD-19, a CORD-19 SDK would be beneficial because it would integrate *multiple* libraries into a single package, designed to be easily interoperable. In particular, these libraries would be implemented in a manner which prioritizes rapid development: the SDK would comprise a *standalone* and *self-contained* development environment with minimal external dependencies. This priority would extend also to software tools that would be bundled together with the new code libraries. These software tools are discussed next.

II The Software Application Package within the Proposed SDK

In addition to the code libraries described above, whose purpose would be to manipulate CORD-19 data to prepare for text mining and data mining operations, our proposed SDK would bundle numerous applications used for database storage, data visualization, and scripting. The goal of this application package would be to provide researchers with a self-contained computing platform optimized for scientific research and findings related to COVID-19. The components within this application package would be selected with an emphasis on tools that could be distributed in source-code fashion, and then compiled within the SDK's development framework with few, if any, external dependencies. In short, the SDK would try to eliminate almost all scenarios where programmers would need to perform a "system install"; for the most part, the entire computing platform (including scripting and database capabilities) could be compiled from source "out-of-the-box". The SDK would also modify the applications included in the package (e.g., embedding plugins to enable the applications to share data amongst themselves) so as to enhance their interoperability and their usefulness for COVID-19 research.

The applications bundled with the SDK would likely include the following components:

- XPDF: A PDF viewer for reading full-text articles (augmented with CORD-19 features, such as integration with biomedical ontologies);
- AngelScript: An embeddable scripting engine that could be used for analytic processing of data generated by text and data mining operations on CORD-19 (see [6]);
- WhiteDB: A persistent database engine that supports both relational and NoSQL-style architectures (see [10]);
- IQmol: Molecular Visualization software that can be used to study chemical data presented in formats such as PDB which are employed by some COVID-19 data sets;
- MeshLab: A general-purpose 3D graphics viewer;
- UDPipe: a C++ library for manipulating CoNLL data;
- LaTeXML: a LaTeX-to-XML converter;
- PositLib: a library for use in high-precision computations based on the "Universal Number" format, which is more accurate than traditional floating-point encoding in some scientific contexts (see [4]).

It is worth noting that a data-mining platform requires *machine-readable* open-access research data (which is a more stringent requirement than simply pairing publications with data that can only be understood by domain-specific software). For example, radiological imaging can be a source of COVID-19 data insofar as patterns of lung scarring, such as "ground-glass opacity," are a leading indicator of the disease. Consequently, diagnostic images of COVID-19 patients are a relevant kind of content for inclusion in a COVID-19 data set (see [11] as a case-study). However, diagnostic images are not in themselves "machine readable." When medical imaging is used in a quantitative context (e.g., applying Machine Learning for diagnostic pathology), it is necessary to perform Image Analysis to convert the raw data — in this case, radiological graphics — into quantitative aggregates. For instance, by using image segmentation to demarcate geometric boundaries one is able to define diagnostically relevant features (such as opacity) represented as a scalar field over the segments. In short, even after research data is openly published, it may be necessary to perform additional analysis on the data for it to be a full-fledged component of a machine-readable information space.[1] To deal with this sort of situation, our proposed SDK would include a *procedural data-modeling vocabulary* that would both identify the interrelationships between data representations and define the workflows needed to convert CORD-19-linked research data into machine-readable data sets.

Another concern in developing an integrated CORD-19 data collection is that of indexing COVID-19 data for both text mining *and* data mining. In particular, our proposed SDK would introduce a system of *microcitations* that apply to portions of manuscripts *as well as* data sets. In the publishing context, a microcitation is defined as a reference to a partially isolated fragment of a larger document, such as a table or figure illustration, or a sentence or paragraph defining a technical term, or (in mathematics) the statement/proof of a definition, axiom, or theorem. In data publishing, "data citations" are unique references to data sets in their entirety or to their smaller parts. A data microcitation is then a fine-grained reference into a data set. For example, a data microcitation can consist of one column in a spreadsheet, one statistical parameter in a quantitative analysis, or "the precise data records actually used in a study" (in the words adopted by the Federation of Earth Science Information Partners to define microcitations; see [9]).

The unique feature we propose for our SDK would be to combine the text-mining and data-mining notions of microcitation into a *unified* framework. In particular, text-based searches against the CORD-19 corpus would try to find matches in the data sets indexed by our SDK alongside matches within textual content. As a concrete example, a concept such as "expiratory flow" appears in CORD-19 both as a table column in research data and as a medical concept discussed in research papers; a unified microcitation framework should therefore map *expiratory flow* as a keyphrase to both textual

[1] This does not mean that diagnostic images (or other graphical data) should not be placed in a data set; only that computational reuse of such data will usually involve certain numeric processing, such as image segmentation. Insofar as this subsequent analysis is performed, the resulting data should wherever possible be added to the underlying image data as a supplement to the data set.

locations and data set parameters. Similarly, a concept such as *2'-C-ethynyl* (mentioned earlier, in the context of transcription errors) should be identified both as a phrase in article texts and as a molecular component present within compounds whose scientific properties are investigated through **CORD-19** research data. In so doing, a search for this concept would then trigger both publication and data-set matches at the same time.

III The AXF (Annotation Exchange) Format

Conceptually, **AXF** is intended for document preparation as well as for text mining. As a rule, annotations deliberately introduced by authors or editors are more likely to be accurate than annotations which depend on Machine Learning or Natural Language Processing. As a result, the best paradigm for machine-readable document corpora (that are well-suited for text mining purposes) are publications composed in anticipation of archival text-mining requirements. The goal of **AXF** is to facilitate the creation of such archives in the future while *also* supporting text mining technology in the present. These two goals are interrelated, because the data structures supporting human-annotated documents can additionally serve as guidelines for aggregating information gleaned from Natural Language Processing modules.

Producing annotations as part of the document-preparation process represents a relatively minor extension to the tasks of authoring and compositing documents. For example, many text segments that would be annotated as Named Entities — such as acronyms, chemical formulae, technical jargon, etc. — require distinct typesetting rules to visually differentiate them from normal text. In LaTeX, the corresponding commands can then be redefined to output annotation data to an auxiliary file, before applying the relevant typesetting instructions. Similarly, for **XML**-based documents, tag and attribute names may be used to isolate charater sequences which are candidates for annotation. Manuscript composition rules also regulate document structure in ways that promote trivial pattern extraction: paragraphs are always denoted by tags or commands, and sentence boundaries are identified by bare punctuation (in well-formed LaTeX, for instance, periods which are *not* punctuation markers have distinct kerning rules and therefore should be notated with distinct commands). Consequently, when dealing with either LaTeX or **XML**, relatively trivial authoring or editing paradigms can be readily applied in order to generate highly structured documents with built-in sentence and Named Entity demarcations.

The **AXF** format is built around data structures that may be produced automatically by pre- and post-processing manuscripts which adhere to certain simple conventions. In particular, these data structures assert the character indices, paragraph and sentence ids, page numbers, and **PDF** page/viewport coordinates for Named Entities (as well as quotations, citations,[2] hyperref links, equations, and other semantically consequential locations in publication content). This degree of semantic detail is possible when text-mining operations are anticipated during the publication process. Of course, **NLP**-based text-mining technology is needed for the relatively less-well-structured publications which constitute most of our digital ecosystem. With the manuscript-based genre of annotation — produced directly from pre-publication manuscripts — serving as a foundation, **AXF**, accordingly, generalizes to include notation related to text and data mining **API**s and file formats. For example, an **AXF** representation can decribe requests and responses against the BeCAS or NextBio **API**s, or encapsulate a file in formats such as **CoNLL** or **PMML** (Predictive Model Markup Language).

Operationally, **AXF** is modeled most directly on the BeCAS **API** [8] and the Linguistic Annotation Framework (**LAF**).[3] In particular, the **AXF** toolkit includes a command-line tool to "query" manuscripts using an interface based on BeCAS. Moreover, **AXF** uses a two-tier node structure similar to **LAF**; at one level is an extensible text-encoding methodology (disussed in the next paragraph), while a higher level defines annotations in terms of directed hypergraphs. Programmatically, **AXF** aims in the canonical case for a level of detail intermediate between linked-data-oriented projects like Web Annotations (which tend to focus mostly on isolatable semantic resources such as citations and named entities) and **NLP**-oriented paradigms such as **LAF**. That is, **AXF** does not natively serialize fine-grained **NLP** data at the level of individual words (the kind of data asserting semantic and morphosyntactic details: lemmatization, Part of Speech, dependency relations), although it does support queries which return sentences as (unparsed) word-sequences. On the other hand, **AXF** offers some granular information about small-scale linguistic units, such as the role of non-alphanumeric characters, or **PDF** coordinates of sentence start and end points. In short, **AXF** occupies a unique space in the landscape of annotation tools at the intersection of application-development, **NLP**, and document-preparation requirements. Moreover, **AXF** can certainly expand to support new capabilities or resources, based on the practical requirements of projects such as **CORD-19**.

For document preparation, **AXF** is paired with our Hypergraph Text Encoding Protocol (**HTXN**), which provides a canonical character encoding against which annotations can be defined. Within this protocol, an annotation target is a character-

[2] Preprocessing docouments allows fairly detailed citation info to be compiled. For instance, Semantic Scholar or Crossref paper ids can be listed in bibliographies, and citation intent and context can be asserted (in the case of intent, this feature is enabled via a modified `\cite` command with an option for authors to note the relation of the cited work to the current document.

[3] However, **AXF** also supports the representation of data structures used by a variety of data and text mining tools and methodologies, such as **PMML**, **ARFF** (Attribute-Relation File Format), **IeXML**, **CoNLL-U**, OpenAnnotations, and **SciXML**.

index interval in the context of an **HTXN** character stream. On that basis, **HTXN** treats documents as graphs whose nodes are ranges in a character stream, where text can be recovered as an operation on one or more nodes (e.g., the text of a sentence is derived from a pair of nodes representing the sentence's start and end). **HTXN** code-points are distinguished in terms of their semantic role, which may be more granular than their visible appearance — for example, a period glyph is assigned different code-points depending on whether it marks a sentence-ending punctuation, an abbreviation, a decimal point, or part of an ellipsis. Procedures are then implemented to represent text in different formats, such as **ASCII**, Unicode, **XML**, or LaTeX. In contrast to a format such as Web Annotations, any particular human-readable text presentation (including **ASCII**) is considered a *derived* property of the annotation, not a foundational representation.

Both **AXF** and **HTXN** are integrated with tools which have been proposed for the **CORD-19** "application package". In particular, LTS has implemented a modified version of the **XPDF** viewer which extracts embedded files containing **AXF** and **HTXN** data, using this information to make context menus sensitive to the location of text which is an annotation target, as well as to sentence boundaries, in the **PDF** viewport (allowing the reader, for instance, to automatically copy one sentence's text to the clipboard, were the relevant sentence is identified by mouse position). Similarly, LTS has implemented an AngelScript interface which can be used to query manuscripts for **AXF** data. These viewer and query capabilities are provided as Research Object code that may be included in published data sets. Anyone who downloads data sets constructed according to this protocol therefore has access to a document viewer which internally supports **AXF** text-extraction features.

IV Conclusion

The LTS vision of a *standalone* and *self-contained* **Covid-19** data-set collection is consistent with new publishing initiatives such as Research Objects (see [1]) and **FAIR** ("Findable, Accessible, Interoperable, Reusable"; see [12]). Indeed, our **CORD-19 SDK** would function as a macro-scale Research Object, which would be (1) *self-contained* (with few or no external dependencies); (2) *transparent* (meaning that all computing operations should be implemented by source code within the bundle that can be examined as code files and within a debugging session); and (3) interactive (meaning that the bundle does not only include raw data but also software to interactively view and manipulate this data). Research Objects which embrace these priorities attempt to provide data visualization, persistence, and analysis through **GUI**, database, and scripting engines that can be embedded as source code in the Research Object itself. Our proposed **SDK** would be based on the same paradigm, but instead of applying the Research Object model to a single data set, our **SDK** would translate it to a larger data space, viz., a research *archive*. Analogous to code bundled with a single research *object*, an archive-specific **SDK** is then associated with each Research Archive. In the case of **CORD-19**, the archive **SDK** would serve the goal of integrating the information contained in multiple **Covid-19** data sets as well as the entire corpus of **CORD-19** articles.

References

[1] Khalid Belhajjame, *et. al.*, "Workflow-centric research objects: First class citizens in scholarly discourse". https://pages.semanticscholar.org/coronavirus-research

[2] "COVID-19 Open Research Dataset (CORD-19)". 2020. Version 2020-03-13. Retrieved from https://pages.semanticscholar.org/coronavirus-research. Accessed 2020-03-20. doi:10.5281/zenodo.3715506 https://pages.semanticscholar.org/coronavirus-research

[3] Luděk Eyer, *et. al.*, "Nucleoside analogs as a rich source of antiviral agents active against arthropod-borne flaviviruses". https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5890575/

[4] John Gustafson, "Beating Floating Point at its Own Game: Posit Arithmetic", http://www.johngustafson.net/pdfs/BeatingFloatingPoint.pdf

[5] Nancy Ide and Keith Suderman, "GrAF: A Graph-based Format for Linguistic Annotations", https://www.cs.vassar.edu/~ide/papers/LAW.pdf

[6] Andreas Jönsson, "AngelCode Scripting Library", www.AngelCode.com/AngelScript/

[7] Amy Neustein, *et. al.*, "Application of Text Mining to Biomedical Knowledge Extraction: Analyzing Clinical Narratives and Medical Literature", https://www.researchgate.net/publication/262372604_Application_of_Text_Mining_to_Biomedical_Knowledge_Extraction_Analyzing_Clinical_Narratives_and_Medical_Literature

[8] Tiago Nunes, *et. al.*, "BeCAS: biomedical concept recognition services and visualization". https://www.ncbi.nlm.nih.gov/pubmed/23736528

[9] Mark A. Parsons and Ruth Duerr, "Data Identifiers, Versioning, and Micro-citation", https://www.thelancet.com/action/showPdf?pii=S1473-3099%2820%2930086-4

[10] Enar Reilent, "Whiteboard Architecture for the Multi-agent Sensor Systems", https://www.thelancet.com/action/showPdf?pii=S1473-3099%2820%2930086-4

[11] Heshui Shi, *et. al.*, "Radiological findings from 81 patients with COVID-19 pneumonia in Wuhan, China: a descriptive study". https://www.thelancet.com/action/showPdf?pii=S1473-3099%2820%2930086-4

[12] Alina Trifan and José Luís Oliveira, "FAIRness in Biomedical Data Discovery". https://www.researchgate.net/publication/331775411_FAIRness_in_Biomedical_Data_Discovery