SPECIAL SERIES: INFORMATICS TOOLS FOR CANCER RESEARCH AND CARE

original reports

CIViCpy: A Python Software Development and Analysis Toolkit for the CIViC Knowledgebase

Alex H. Wagner, PhD[1,2]; Susanna Kiwala, MS[1]; Adam C. Coffman, BS[1]; Joshua F. McMichael, BS[1]; Kelsy C. Cotto, BS[1]; Thomas B. Mooney, MS[1]; Erica K. Barnell, MD, PhD[1]; Kilannin Krysiak, PhD[3]; Arpad M. Danos, PhD[1]; Jason Walker, MS[1]; Obi L. Griffith, PhD[1,2,4,5]; and Malachi Griffith, PhD[1,2,4,5]

abstract

PURPOSE Precision oncology depends on the matching of tumor variants to relevant knowledge describing the clinical significance of those variants. We recently developed the Clinical Interpretations for Variants in Cancer (CIViC; civicdb.org) crowd-sourced, expert-moderated, and open-access knowledgebase. CIViC provides a structured framework for evaluating genomic variants of various types (eg, fusions, single-nucleotide variants) for their therapeutic, prognostic, predisposing, diagnostic, or functional utility. CIViC has a documented application programming interface for accessing CIViC records: assertions, evidence, variants, and genes. Third-party tools that analyze or access the contents of this knowledgebase programmatically must leverage this application programming interface, often reimplementing redundant functionality in the pursuit of common analysis tasks that are beyond the scope of the CIViC Web application.

METHODS To address this limitation, we developed CIViCpy (civicpy.org), a software development kit for extracting and analyzing the contents of the CIViC knowledgebase. CIViCpy enables users to query CIViC content as dynamic objects in Python. We assess the viability of CIViCpy as a tool for advancing individualized patient care by using it to systematically match CIViC evidence to observed variants in patient cancer samples.

RESULTS We used CIViCpy to evaluate variants from 59,437 sequenced tumors of the American Association for Cancer Research Project GENIE data set. We demonstrate that CIViCpy enables annotation of $>$ 1,200 variants per second, resulting in precise variant matches to CIViC level A (professional guideline) or B (clinical trial) evidence for 38.6% of tumors.

CONCLUSION The clinical interpretation of genomic variants in cancers requires high-throughput tools for interoperability and analysis of variant interpretation knowledge. These needs are met by CIViCpy, a software development kit for downstream applications and rapid analysis. CIViCpy is fully documented, open-source, and available free online.

JCO Clin Cancer Inform 4:245-253. © 2020 by American Society of Clinical Oncology

ASSOCIATED CONTENT

Data Supplement

Author affiliations and support information (if applicable) appear at the end of this article.

Accepted on January 15, 2020 and published at ascopubs.org/journal/cci on March 19, 2020: DOI https://doi.org/10.1200/CCI.19.00127

INTRODUCTION

The use of massively parallel sequencing to profile the molecular composition of human tissues has become increasingly commonplace in the clinical setting to inform diagnosis and therapeutic strategy for patients' tumors.[1,2] This has led to an ever-growing body of biomedical literature describing the impact of tumor variants on disease progression and response to therapy, creating a bottleneck of expert review of relevant literature to construct a clinical report.[3] The Clinical Interpretations for Variants in Cancer (CIViC) community knowledgebase (civicdb.org) is a platform for expert crowdsourcing the clinical interpretation of variants in cancer.[4] To date, CIViC contains 6,471 interpretation evidence records (ie, clinical significance statements extracted from biomedical literature) describing 2,312 variants in 402 genes. Evidence in CIViC is used to construct interpretation assertions of clinical significance (ie, therapeutic, prognostic, diagnostic, or predisposing effects) of gene variants on the basis of published criteria and guidelines for the classification of variant interpretations.[5,6] CIViC evidence and assertions are also linked to data classes describing genes, drugs (if applicable), and diseases, in addition to the myriad supporting data for tracking the provenance and community activity surrounding these concepts and their relationships. These data are released under a Creative Commons public domain attribution (CC0), promoting their redistribution and use in downstream applications.

As a curation platform for the Clinical Genome Resource (ClinGen) Somatic Working Group,[7] CIViC supports the export of generated assertions to ClinVar, in line with existing ClinGen submission practices for

CONTEXT

Key Objective

To develop a software development and analysis toolkit for performant search and inspection of records from the Clinical Interpretations for Variants in Cancer (CIViC) knowledgebase.

Knowledge Generated

We developed the CIViCpy software development kit and analysis toolkit, a software package that enables high-throughput retrieval and inspection of CIViC records. Useful features include precached content hosted live by CIViC, and utilities for exporting CIViC content as variant call format (VCF) files.

Relevance

This work enables (1) building of downstream applications, such as the civic2clinvar ClinVar submission utility; (2) clinical analysis tasks, such as the demonstrated analysis of the American Association for Cancer Research Project GENIE cohort; and (3) CIViC integration into clinical annotation pipelines, via VCF export or direct record annotation with native Python objects.

germline diseases.[8] This was accomplished through the development of the civic2clinvar export utility and Python package, which constructs ClinVar-style submission records from CIViC assertions.[7] In developing civic2clinvar, several issues with building an application from the CIViC database and application programming interface (API) were identified: (1) simplified retrieval and use of CIViC records as native Python objects, (2) routines for local caching of CIViC data for analysis, (3) support for high-throughput queries, and (4) export of CIViC records to established variant representation formats, such as variant call format (VCF).[9]

Here, we describe CIViCpy, a software development kit (SDK) that addresses these needs and enables rapid downstream tool development and analysis by removing the burden of implementing these features in independent applications. We demonstrate the use of the SDK in an associated analysis notebook to evaluate 59,437 tumors from patients cataloged by the American Association for Cancer Research Project GENIE cohort.[10] CIViCpy is open-source, Massachusetts Institute of Technology (MIT) licensed, and readily available for installation on the Python Package Index (PyPI; pypi.org). CIViCpy documentation is available online at civicpy.org.

METHODS

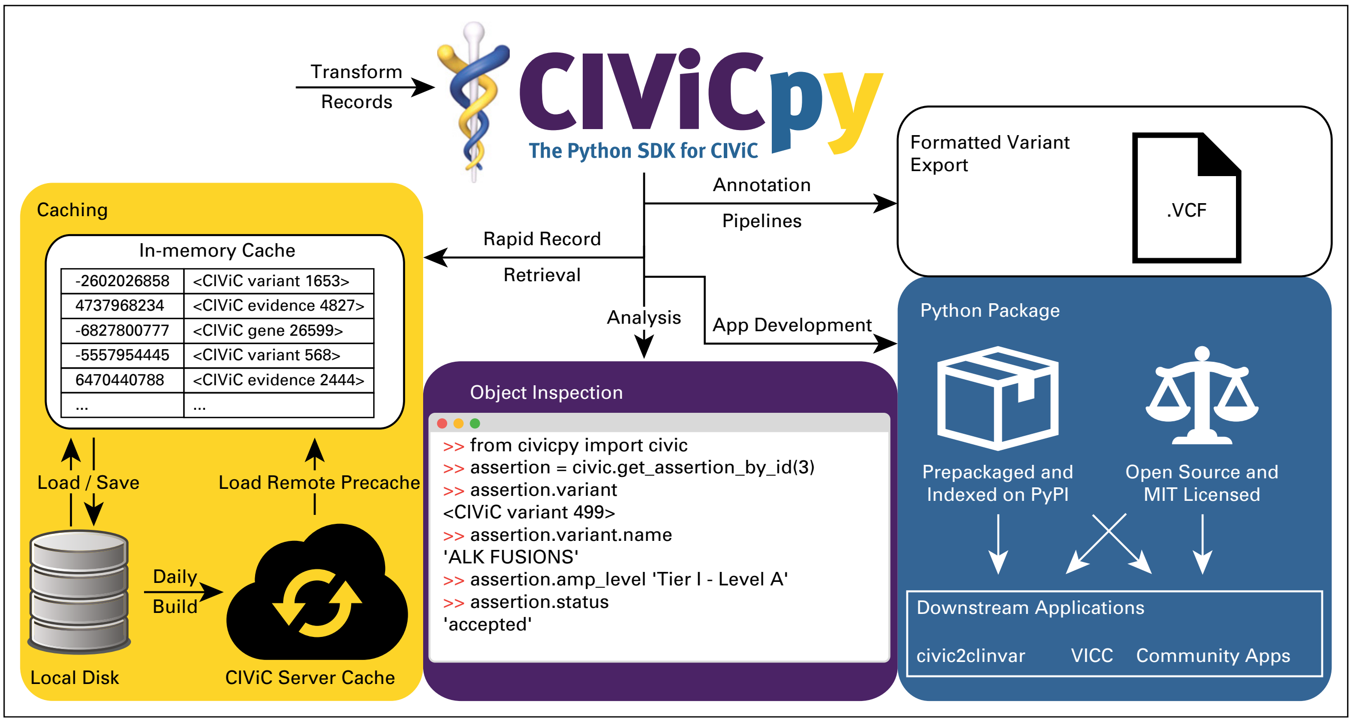
We designed the CIViCpy Python SDK as a standalone package to retrieve the CIViC knowledgebase content and transform responses into Python objects with intuitive structures and interobject linkages. The resulting software is a toolkit to support numerous downstream operations, including exploratory analyses, variant annotation, and application development (Fig 1). Here, we describe the optimizations and design choices made to construct CIViCpy.

CIViCpy Objects

The primary data class in CIViCpy is the CivicRecord. This class provides the framework for all first-class entities in CIViC: Genes, Variants, Variant Groups, Evidence, and Assertions. First-class entities are delineated from other object classes in CIViC by the combination of persistent public identifiers, dedicated API end points for returning object details, and tracked provenance (Table 1). Provenance tracking records the history of all actions taken on the object as part of the CIViC curation cycle: object submission, revisions, and editor approval. We also create CivicRecord objects from CIViC Sources, Users, and Organizations, despite their lack of provenance tracking; CivicRecord objects only require that a CIViC class is identifiable and has supporting API end points. Documentation for each of the CivicRecord subclasses can be found online at http://bit.ly/civicrecord-types.

The CIViCpy CivicAttribute is a data class for representing composite data entities not captured by CivicRecord. This includes composite entities with nested or list attributes (eg, *diseases, coordinates, or variant_aliases*), as opposed to primitive, single-valued entities (eg, *description, allele_registry_id,* or *evidence_direction*). CivicAttribute inherits from CivicRecord but is not indexed and accordingly overrides many of the features of its parent class. Importantly, CivicAttribute is not cached (see Caching) except as a linked object to other (non-CivicAttribute) CivicRecord objects, and cannot be retrieved independently.

One of the strengths of the CivicRecord class is the ability to dynamically evaluate nested CivicRecord objects. A Variant, for instance, may have multiple associated Evidence records, each of which may have a source linked to multiple Evidence records describing other Variants. A CivicRecord will automatically link nested objects; consequently, one can chain through linked objects when analyzing CIViC records to efficiently get to values of interest, such as evaluating the Association for Molecular Pathology/ASCO/College of American Pathologists somatic variant classification[6] for a CIViC Assertion (Fig 1, Object Inspection). Therefore, when evaluating evidence for a variant using CIViCpy, the associated Evidence



-2602026858	<CIViC variant 1653>
4737968234	<CIViC evidence 4827>
-6827800777	<CIViC gene 26599>
-5557954445	<CIViC variant 568>
6470440788	<CIViC evidence 2444>
...	...

```
>> from civicpy import civic
>> assertion = civic.get_assertion_by_id(3)
>> assertion.variant
<CIViC variant 499>
>> assertion.variant.name
'ALK FUSIONS'
>> assertion.amp_level 'Tier I - Level A'
>> assertion.status
'accepted'
```

FIG 1. The CIViCpy software development kit (SDK). CIViCpy is a Python SDK and analysis toolkit for Clinical Interpretation of Variants in Cancer (CIViC). The primary function of CIViCpy is to extract records from CIViC, convert them into linked Python objects, and provide useful tools and features for exploring and analyzing those records. Caching: An important feature of CIViCpy is rapid retrieval via caching, with support for saving and loading CIViC record caches. CIViCpy is used to build caches hosted on civicdb.org, which can be automatically downloaded to other CIViCpy clients for daily snapshots of all content in CIViC. Object Inspection: CIViCpy enables dynamic evaluation of linked CIViC records through Python dot notation. Here, we see an example of the CIViCpy library in use to explore details about an Association for Molecular Pathology Tier I somatic classification Assertion about ALK fusion transcripts. Python Package: CIViCpy is available on the Python Package Index (PyPI), and is fully open source and permissively licensed for downstream applications. Formatted Variant Export: CIViCpy supports the export of CIViC records into the established variant call format for use in annotation pipelines and existing bioinformatics tools. App, application; MIT, Massachusetts Institute of Technology; VICC, Variant Interpretation for Cancer Consortium; VCF, variant call format.

objects are returned rather than a list of evidence identifiers. Evidence, in turn, will link to other CivicRecord objects (eg, Assertion, Source), which can also be explored. An example of exploring Assertions linked to a Variant using CIViCpy is provided in the Data Supplement (see the CIViCpy Objects section).

Querying CIViC

CIViC is built on a segregated server/client architecture, where all functionality of the CIViC Web client is managed through RESTful API calls to the underlying Rails server. CIViCpy leverages this architectural design to post complex queries to the CIViC advanced search API end points. This enables high-throughput searches for records of interest, including full data set requests. CIViC full data sets include Evidence and Assertions in multiple states of the CIViC review cycle: those that have been editor-reviewed and approved (accepted), those that are pending review (submitted), and those that have been rejected for inclusion in CIViC. Because the CIViC full data set contains rejected and submitted Evidence items and Assertions that have not been approved by CIViC editors, these records may be inaccurate, in a partial state, or incongruent with the CIViC knowledge model.

CivicRecord objects may be retrieved by the corresponding get_all functions (eg, *get_all_variants(), get_all_assertions()*). These functions may optionally be passed a parameter for explicitly including only objects of a given status. For example, a user may request only evidence that has been accepted or submitted (and exclude any rejected evidence). Although these behaviors are readily reproducible by users without leveraging this feature (eg, through inspection of *evidence.status*), we expect most downstream workflows would desire to include only accepted and/or submitted evidence, and so we have provided this functionality as a convenience. An example of filtering evidence by status using CIViCpy is provided in the Data Supplement (see the Filtering Evidence section).

Caching

CIViCpy also is a standalone component for services intended to perform large-volume operations on the CIViC knowledgebase. A key design consideration, therefore, is the local caching of CIViC content for quick retrieval and

TABLE 1. CIViC Class Entities

CIViC Object	Provenance Tracking	Persistent CIViC ID	Detailed Record End Point	CIViCpy Class
First-class entities				
Genes	Y	Y	Y	CivicRecord
Variants	Y	Y	Y	CivicRecord
Variant groups	Y	Y	Y	CivicRecord
Evidence	Y	Y	Y	CivicRecord
Assertions	Y	Y	Y	CivicRecord
Second-class entities				
User	N	Y	Y	CivicRecord
Organization	N	Y	Y	CivicRecord
Sources	N	N	Y	CivicRecord
Country	N	N	N	CivicAttribute
Drugs	N	N	N	CivicAttribute
Diseases	N	N	N	CivicAttribute
Phenotypes	N	N	N	CivicAttribute

Abbreviations: CIViC, Clinical Interpretations for Variants in Cancer; ID, identifier.

local indexing operations. When loaded, each cached CivicRecord is assigned a unique key using the native Python hash function, which is recomputed on loading a stored cache from disk (by design, the hash seed changes with each Python session). This hash is computed on the CivicRecord *type* and *id* values, such that user-generated partial CivicRecords with these minimal values provide the same hash (and are treated as equivalent to) complete CivicRecords.

Each cache also maintains a timestamp of when the cache was last generated. This information is used when loading the cache from file to determine whether a fresh cache needs to be built or retrieved from a remote source. CIViCpy will expire a cache 7 days (or after user-configurable length of time) after it is initially built and will retrieve the newest (nightly) cache from the CIViC live server. The local cache can also be manually updated from the command line with the civicpy update utility.

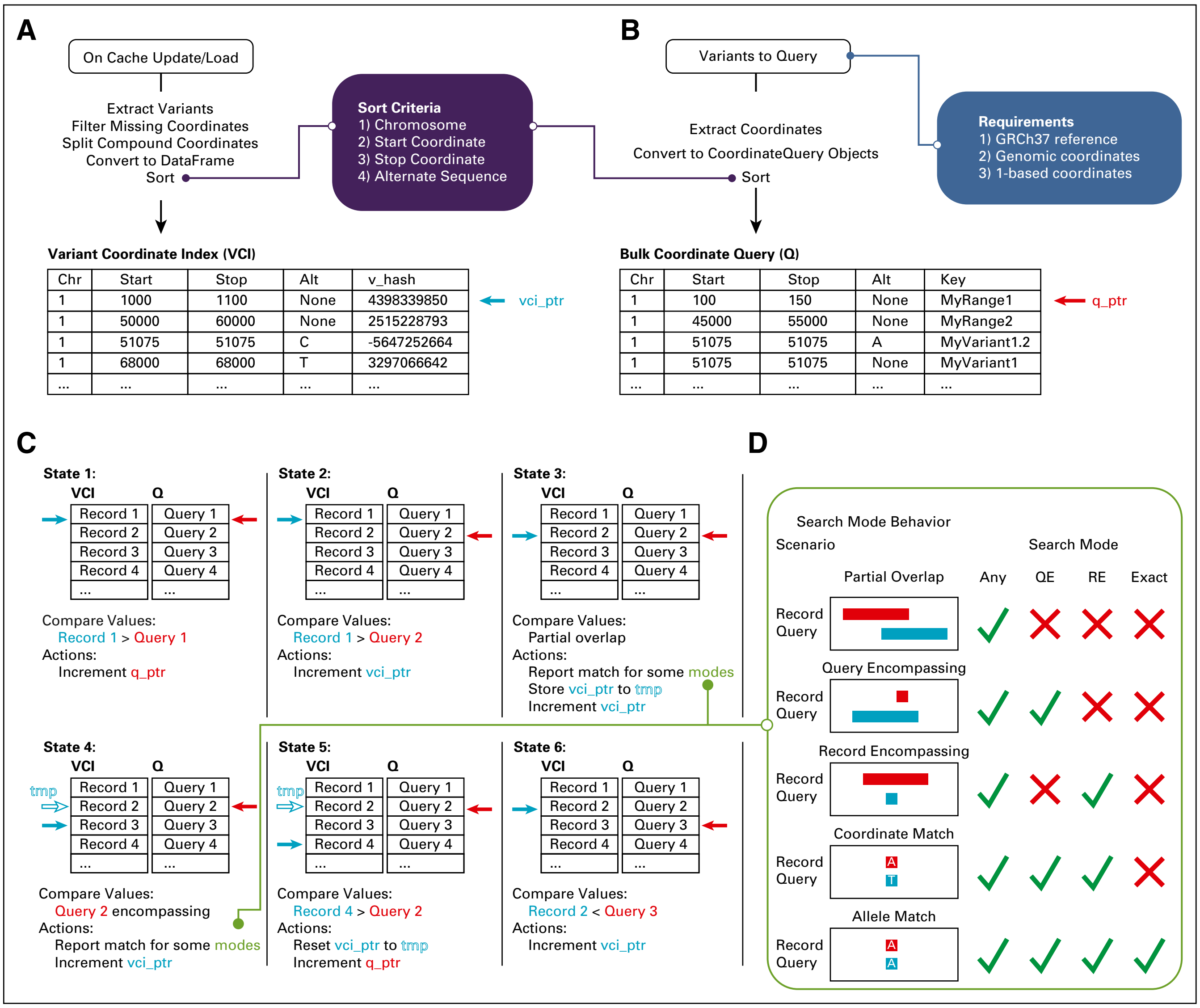
Variant Coordinate Search

When loading all variants from CIViC, a sorted variant coordinate index is also constructed after cache generation to enable coordinate search and lookup strategies (Fig 2A). A similarly sorted list of CoordinateQuery objects represent variants to query, such as those observed in a patient's tumor (Fig 2B). The index and CoordinateQuery objects are used by the search algorithm for high-throughput searches (Fig 2C). Importantly, the search algorithm supports the notion of variant ranges for CIViC records and queries, and provides several search modes to accommodate varying sensitivity and specificity tradeoffs (Fig 2D).

Variant Exports

CIViCpy enables the export of CIViC Variant records and their associated Evidence Items and Assertions into the VCF (Fig 1) by providing a VCFWriter class. After instantiating a VCFWriter object, a Variant record may be added to it for future output by calling the *addrecord()* function. The *addrecord()* function supports various variant types, depending on the curated coordinates available for a Variant. All Variants require the chromosome name and start position. For single-nucleotide variants and complex variants, the reference sequence and altered sequence information also must be available. By contrast, insertions require only variant sequence information and deletions require only reference sequence information. Variants that do not meet these minimum requirements will not be added and a warning message is emitted instead. Fusions and other variants with a second set of coordinates are currently not supported. To verify whether a Variant can be added to a VCFWriter object, the convenience method *is_valid_for_vcf()* can be called on a Variant object before calling *addrecord()*. More information about variant types that may lack reference and variant sequence information (eg, fusions) can be found at bit.ly/civic-coordinates. Those variants that cannot be exported into the VCF format are still retrievable as CIViCpy records. Once all desired variants are added to the VCFWriter object, *writerecords()* needs to be called to write the VCF file. An example of using CIViCpy to generate a CIViC VCF file is provided in the Data Supplement (see the Exporting to VCF section).

The variants added to the VCFWriter object are written to the VCF file, one VCF record for each Variant object. If two Variant objects share the same chromosome, start position, and reference allele(s), they will not be combined into one VCF record but instead will be written as separate VCF records. Additional CIViC data are added to the VCF as annotations to the CSQ (consequence) INFO field (Table 2). CIViC Evidence items and Assertions linked to the Variant


A
On Cache Update/Load
Extract Variants
Filter Missing Coordinates
Split Compound Coordinates
Convert to DataFrame
Sort
Sort Criteria
1) Chromosome
2) Start Coordinate
3) Stop Coordinate
4) Alternate Sequence
Variant Coordinate Index (VCI)
vci_ptr
B
Variants to Query
Extract Coordinates
Convert to CoordinateQuery Objects
Sort
Requirements
1) GRCh37 reference
2) Genomic coordinates
3) 1-based coordinates
Bulk Coordinate Query (Q)
q_ptr
C
State 1: Compare Values: Record 1 > Query 1 Actions: Increment q_ptr
State 2: Compare Values: Record 1 > Query 2 Actions: Increment vci_ptr
State 3: Compare Values: Partial overlap Actions: Report match for some modes Store vci_ptr to tmp Increment vci_ptr
State 4: Compare Values: Query 2 encompassing Actions: Report match for some modes Increment vci_ptr
State 5: Compare Values: Record 4 > Query 2 Actions: Reset vci_ptr to tmp Increment q_ptr
State 6: Compare Values: Record 2 < Query 3 Actions: Increment vci_ptr
VCI Q Record 1 Record 2 Record 3 Record 4 ... Query 1 Query 2 Query 3 Query 4 ...
tmp
D
Search Mode Behavior
Scenario
Search Mode
Any QE RE Exact
Partial Overlap ✓ ✗ ✗ ✗
Query Encompassing ✓ ✓ ✗ ✗
Record Encompassing ✓ ✗ ✓ ✗
Coordinate Match ✓ ✓ ✓ ✗
Allele Match ✓ ✓ ✓ ✓
Record Query

Variant Coordinate Index (VCI)

Chr	Start	Stop	Alt	v_hash
1	1000	1100	None	4398339850
1	50000	60000	None	2515228793
1	51075	51075	C	-5647252664
1	68000	68000	T	3297066642
...	...	...	...	...

Bulk Coordinate Query (Q)

Chr	Start	Stop	Alt	Key
1	100	150	None	MyRange1
1	45000	55000	None	MyRange2
1	51075	51075	A	MyVariant1.2
1	51075	51075	None	MyVariant1
...	...	...	...	...

FIG 2. Variant coordinate search with CIViCpy. (A) On updating or loading the in-memory cache, all variant records are extracted and converted into a sorted Variant Coordinate Index (VCI). Variants missing coordinate values are excluded, and variants with compound coordinates (eg, fusion variants) have the coordinates split into distinct records. Each coordinate is sorted by chromosome, start coordinate, stop coordinate, and alternate sequence (see the purple box). The VCI also contains a reference to the cache key for the corresponding variant. The vci_ptr is a reference to a record of the VCI. (B) CoordinateQuery objects are used to query against the VCI and should match the coordinate system requirements from CIViC (see the blue box). These objects contain an optional key field for user reference. When searching the knowledgebase for several variants, CoordinateQuery objects should be presorted by the same procedure for sorting the VCI. The q_ptr serves an analogous role to the vci_ptr for the variant queries. (C) Starting at the first sorted VCI and Query record, searches will increment the smaller of the vci_ptr or q_ptr until an overlapping coordinate range is identified. When overlaps occur, matches are reported on the basis of the search model specified. The vci_ptr is restored once all overlapping records for a query are evaluated. (D) Several search models exist for this algorithm, including a highly sensitive Any search, a conservative Exact search, and two intermediate modes. Overlap scenarios and report behavior for each scenario are presented. ✓, a scenario is reported; ✕, a scenario is not reported. QE, query encompassing; RE, record encompassing.

are added to the CSQ field with one CSQ entry for each Evidence item and/or Assertion. Whether a specific CSQ entry reflects an Evidence item or an Assertion is determined by the CIViC Entity Type CSQ field. To differentiate special characters in the field values from field delimiters, spaces are replaced with underscores and other special characters are hex encoded. By using the CSQ field for annotations, the resulting VCF is compatible for import into Google BigQuery (git.io/bigquery-variant-annotation).

A command line utility, *civicpy create-vcf*, allows users to create a VCF file of all supported CIViC variants. The *-i/–include-status* required parameter will restrict the annotations to only those Evidence Items and Assertions that

TABLE 2. Variant Call Format CSQ Field Attributes

CSQ Field	Description	Compound Field[a]
Allele	Alternate allele	N
Consequence	CIViC sequence ontology variant types for this variant	Y
SYMBOL	HGNC gene symbol for the gene associated with this variant	N
Entrez Gene ID	Entrez gene ID for the gene associated with this variant	N
Feature_type	“transcript”	N
Feature	The Ensembl ID for the CIViC representative transcripts of this variant	N
HGVSc	Variant representation using HGVS notation (DNA level), corresponding to the Feature	N
HGVSp	Variant representation using HGVS notation (protein level), corresponding to the Feature	N
CIViC Variant Name	The CIViC variant name of this variant	N
CIViC Variant ID	The CIViC internal ID for this variant	N
CIViC Variant Aliases	CIViC aliases for this variant	Y
CIViC HGVS	CIViC HGVS strings for this variant	Y
Allele Registry ID	The allele registry ID for this variant	N
ClinVar IDs	ClinVar IDs associated with this variant	Y
CIViC Variant Evidence Score	The CIViC evidence score for this variant	N
CIViC Entity Type	The type of entity being annotated, either “evidence” or “assertion”	N
CIViC Entity ID	The CIViC internal ID for the entity being annotated	N
CIViC Entity URL	The CIViC direct URL to the entity being annotated	N
CIViC Entity Source	For evidence entities, the ID of the publication used to create the evidence including the source type in the format “sourceId_(sourceType)”	N
CIViC Entity Variant Origin	The variant origin of the entity being annotated, either “Somatic,” “Rare Germline,” “Common Germline,” “Unknown,” or “N/A”	N
CIViC Entity Status	The status of the CIViC entity being annotated, either “submitted,” “accepted,” or “rejected”	N

Abbreviations: CIViC, Clinical interpretation of Variants in Cancer; CSQ, consequence; HGNC, HUGO Gene Nomenclature Committee; HGVS, Human Genome Variation Society; ID, identifier; N/A, not applicable.

[a]Compound fields contain multiple values and use the ampersand (&) character to delineate values.

match the given status(es). If a variant does not have any supporting Evidence Items or Assertions with the required statuses, it will not be included in the VCF.

Software Engineering and Availability

The CIViCpy codebase is hosted publicly on GitHub (git.io/civicpy). The test suite is implemented using the pytest framework and GitHub integration tests are run using travis-ci (travis-ci.org). Coveralls (https://coveralls.io/) is used to track test coverage (77%). Code changes are integrated using GitHub pull requests (https://github.com/griffithlab/civicpy/pulls). Feature additions, user requests, and bug reports are managed using GitHub issue tracking (https://github.com/griffithlab/civicpy/issues). Collectively, these features enable robust community development and facilitate adoption of the CIViCpy SDK.

User documentation is written using reStructuredText markup language and the Sphinx documentation framework (sphinx-doc.org). Documentation is hosted on Read the Docs (readthedocs.org) and can be viewed at civicpy.org. This documentation serves as both a “quick start” guide and a detailed reference for developers and bioinformaticians.

This project is licensed under the MIT License (https://opensource.org/licenses/MIT). CIViCpy has been packaged and uploaded to the PyPI under the civicpy package name and can be installed by running the pip install civicpy command. Installation requires a Python version 3.7 environment. Releases are also made available on GitHub (https://github.com/griffithlab/civicpy/releases).

GENIE Analysis

To evaluate the performance of CIViCpy in annotating patient data, we performed a demonstrative analysis in a Jupyter Notebook, available on the CIViCpy GitHub repository (git.io/civicpy-genie). The Project GENIE[10] version 5.0 extended mutations file, which describes 445,655 variants across 59,437 patient tumors, was downloaded from https://www.synapse.org/#!Synapse:syn17394041. Coordinates from the reported variants were extracted and each coordinate set was tagged using the corresponding tumor sample barcode.

The extracted coordinates were then transformed into a sorted list of CIViCpy CoordinateQuery objects, which were passed to the bulk-query search method using an exact search strategy (Fig 2D). Match results and query times were recorded for the full set of GENIE variants in addition to timings from exponentially increasing subsets from one to 300,000. Match results from an anticonservative search strategy, which allowed for any coordinate overlap (Fig 2D), were also recorded.

Match results from the full set of variants were grouped by tumor identifier and summarized by counts of Exact, Any, and no matches. In addition, tumors for which no variations were reported for querying were also summarized. Finally, we grouped tumors on the number of variants matching CIViC evidence by Exact search, and summarized the highest level of evidence found across the tumors in those groups.

RESULTS

GENIE Tumor Variant Results

Using the Exact search strategy, CIViCpy successfully matched CIViC evidence to 7.6% (n = 34,642) of GENIE variants, and using the *Any* strategy, an additional 42.9% of variants (n = 195,349) were matched (Fig 3A). Furthermore, 46.3% of tumors (n = 27,545) had at least one Exact match to a reported variant and an additional 40.2% of tumors (n = 23,925) had at least one Any match. Notably, 8.7% of tumors (n = 5,200) in the cohort had no reported variants to query against the knowledgebase. We evaluated the highest CIViC evidence level reported for the 27,545 tumors with matched evidence, and found 14.0% of those tumors (n = 3,852) matched to a CIViC Validated Association (Level A). An additional 69.3% of tumors (n = 19,098) matched to Clinical Evidence (Level B). Far fewer tumors had Case Study (Level C, 11.8%; n= 3,248), Preclinical (Level D, 3.9%; n = 1,066), or Inferential (Level E, 1.0%; n = 281) as the highest-level evidence match (Fig 3B). In total, 38.6% (n = 22,950) of all GENIE tumors (including those without reported variants) had at least one variant that matched to Level A or B evidence. Tumors from the cohort had an average of 6.88 (median, 4) variants reported, and although most tumors (53.7%; n = 31,892) did not Exact match to CIViC variants, many tumors matched one (35.4%; n = 21,026), two (9.4%; n = 5,592), three (1.3%; n = 796), or more (0.2%; n = 131) CIViC variants.

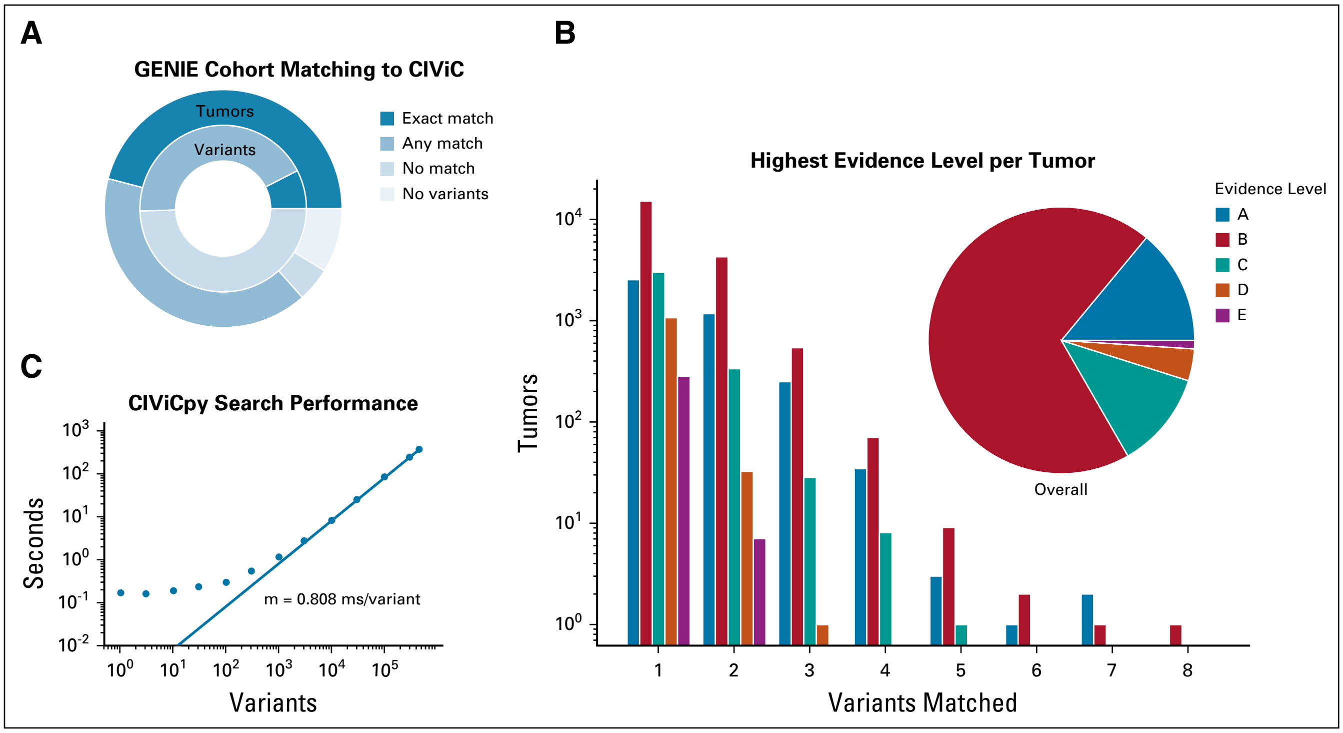


FIG 3. Variant coordinate search with CIViCpy. (A) Querying GENIE variants against the CIViC knowledgebase results in 7.6% (n = 34,642) of variant (n = 445,655) and 46.3% (n = 27,545) of tumors (n = 59,437) with Exact matches to CIViC variants. Allowing for Any matches, these values increase to 50.5% of variants (n = 229,991) and 86.6% of tumors (n = 51,470). A small percentage (8.7%; n = 5,200) of all tumors had no reported variants with which to search. (B) Tumors were grouped by the number of variants that Exact matched CIViC records. For each tumor, the highest level of evidence supporting the matched variants was counted. We observed that 83.3% of tumors (n = 22,950) with Exact matching variants (which equates to 38.6% of all tumors) were linked to CIViC Validated (Level A) or Clinical (Level B) evidence. (C) Response time of CIViCpy bulk queries against the CIViC knowledgebase. As the number of variants queried increase > 1,000, response time is linear to the number of queries, with an increase in overall time of 0.808 ms/variant. CIViC, Clinical Interpretations for Variants in Cancer.

CIViCpy Search Performance

CIViCpy uses a search strategy (Fig 2C) designed to scale linearly with input query size. We evaluated the performance of CIViCpy annotation of the GENIE cohort and found that queries with $\geq$ 1,000 variants scale linearly with input size, increasing total search time by approximately 0.808 ms/variant (Fig 3C). Queries with $<$ 1,000 variants exhibit higher performance and are returned in $<$ 1 second. We annotated the entire GENIE data set (n = 445,655 variants) in 369 seconds at a rate of 1,236 variants/s.

DISCUSSION

CIViCpy is an SDK and high-throughput analysis toolkit for exploring and analyzing content within the CIViC knowledgebase. CIViCpy has tools for object inspection, with convenient features for record retrieval and search. Variant annotation through CIViCpy is demonstrated to handle variant searches at 1,236 variants/s through the provided coordinate search methods. In addition, the SDK provides convenient tools for exporting CIViC content to VCF for integration in external annotation pipelines and tools.

The CIViCpy SDK has demonstrated utility in downstream applications, including the previously published civic2clinvar utility[7] and the Variant Interpretation for Cancer Consortium.[11] Additional features to improve CIViCpy are already being planned, including extensions for other variant export formats such as the Browser Extensible Data[12] and the Global Alliance for Genomics and Health VR specification.[13] These extensions will also support export of variant types beyond the single-nucleotide variants and insertions/deletions currently supported by our VCF export utility. In addition, we are planning to develop utilities to allow users to annotate their own VCFs with CIViC data. We are also developing strategies to incorporate CIViC Drug, Disease, and Phenotype entities as full CivicRecord objects.

Finally, we have provided all source code for CIViCpy (git.io/civicpy) and the analyses in this manuscript (git.io/cpy-genie) in a public repository under the permissive MIT license and have uploaded CIViCpy distributions to the PyPI for ease of installation. The permissive licensing and easy installation through PyPI allow for rapid integration into existing analytical workflows. See our documentation and project homepage at civicpy.org for more details.

AFFILIATIONS

[1]McDonnell Genome Institute, Washington University School of Medicine, St Louis, MO
[2]Department of Medicine, Washington University School of Medicine, St Louis, MO
[3]Department of Pathology and Immunology, Washington University School of Medicine, St Louis, MO
[4]Siteman Cancer Center, Washington University School of Medicine, St Louis, MO
[5]Department of Genetics, Washington University School of Medicine, St Louis, MO

Preprint version available on bioRxiv.

CORRESPONDING AUTHOR

Obi L. Griffith, PhD, McDonnell Genome Institute, Washington University School of Medicine, Campus Box 8501 4444 Forest Park Ave, St Louis, MO 63108; email: obigriffith@wustl.edu

EQUAL CONTRIBUTION

A.H.W. and S.K. contributed equally to this work. O.L.G. and M.G. contributed equally to this work.

SUPPORT

A.H.W. was supported by a fellowship from the National Cancer Institute (NCI; National Institutes of Health [NIH] Grant No. F32CA206247), a National Human Genome Research Institute (NHGRI) career development award (NIH Grant No. K99HG010157), and the Foundation for Barnes-Jewish Hospital (Award Group 615735). M.G. was supported by a career development award from the NHGRI (NIH Grant No. R00HG007940). A.H.W., S.K., A.C.C., J.F.M., E.K.B., K.K., A.M.D., O.L.G., M.G., and the CIViC knowledgebase were supported by the NCI (NIH Grant No. U01CA209936 and U24CA237719). VCF export development was supported by Google Genomics. This research was also supported by a Cancer Moonshot funding opportunity, specifically an Activities to Promote Technology Research Collaborations for Cancer Research (Administrative Support) award to O.L.G. (Grant No. U01CA209936-S).

AUTHOR CONTRIBUTIONS

Conception and design: Alex H. Wagner, Susanna Kiwala, Adam C. Coffman, Joshua F. McMichael, Kelsy C. Cotto, Erica K. Barnell, Jason Walker, Obi L. Griffith, Malachi Griffith
Financial support: Alex H. Wagner, Obi L. Griffith, Malachi Griffith
Administrative support: Jason Walker, Malachi Griffith
Collection and assembly of data: Alex H. Wagner, Susanna Kiwala, Kilannin Krysiak, Malachi Griffith
Data analysis and interpretation: Alex H. Wagner, Susanna Kiwala, Thomas B. Mooney, Arpad M. Danos
Manuscript writing: All authors
Final approval of manuscript: All authors
Accountable for all aspects of the work: All authors

AUTHORS' DISCLOSURES OF POTENTIAL CONFLICTS OF INTEREST

The following represents disclosure information provided by authors of this manuscript. All relationships are considered compensated unless otherwise noted. Relationships are self-held unless noted. I = Immediate Family Member, Inst = My Institution. Relationships may not relate to the subject matter of this manuscript. For more information about ASCO's conflict of interest policy, please refer to www.asco.org/rwc or ascopubs.org/cci/author-center.
Open Payments is a public database containing information reported by companies about payments made to US-licensed physicians (Open Payments).

Erica K. Barnell
Employment: Geneoscopy
Stock and Other Ownership Interests: Geneoscopy
Patents, Royalties, Other Intellectual Property: Inventor on intellectual property in start-up company (Geneoscopy).
Travel, Accommodations, Expenses: Geneoscopy

Kilannin Krysiak
Consulting or Advisory Role: Gerson Lehrman Group

No other potential conflicts of interest were reported.

REFERENCES

1. Freedman AN, Klabunde CN, Wiant K, et al: Use of next-generation sequencing tests to guide cancer treatment: Results from a nationally representative survey of oncologists in the United States. JCO Precision Oncology 2:1-13, 2018
2. Lander ES: Initial impact of the sequencing of the human genome. Nature 470:187-197, 2011
3. Good BM, Ainscough BJ, McMichael JF, et al: Organizing knowledge to enable personalization of medicine in cancer. Genome Biol 15:438, 2014
4. Griffith M, Spies NC, Krysiak K, et al: CIViC is a community knowledgebase for expert crowdsourcing the clinical interpretation of variants in cancer. Nat Genet 49:170-174, 2017
5. Richards S, Aziz N, Bale S, et al: Standards and guidelines for the interpretation of sequence variants: A joint consensus recommendation of the American College of Medical Genetics and Genomics and the Association for Molecular Pathology. Genet Med 17:405-424, 2015
6. Li MM, Datto M, Duncavage EJ, et al: Standards and guidelines for the interpretation and reporting of sequence variants in cancer: A joint consensus recommendation of the Association for Molecular Pathology, American Society of Clinical Oncology, and College of American Pathologists. J Mol Diagn 19:4-23, 2017
7. Danos AM, Ritter DI, Wagner AH, et al: Adapting crowdsourced clinical cancer curation in CIViC to the ClinGen minimum variant level data community-driven standards. Hum Mutat 39:1721-1732, 2018
8. US Food and Drug Administration: Genetic Database Recognition Decision Summary for ClinGen Expert Curated Human Variant Data. Submission No: Q181150. https://www.fda.gov/media/119313/download
9. Danecek P, Auton A, Abecasis G, et al: The variant call format and VCFtools. Bioinformatics 27:2156-2158, 2011
10. AACR Project GENIE Consortium: AACR Project GENIE: Powering precision medicine through an international consortium. Cancer Discov 7:818-831, 2017
11. Wagner AH, Walsh B, Mayfield G, et al: A harmonized meta-knowledgebase of clinical interpretations of cancer genomic variants. bioRxiv 366856. doi: https://doi.org/10.1101/366856.
12. Karolchik D, Hinrichs AS, Furey TS, et al: The UCSC Table Browser data retrieval tool. Nucleic Acids Res 32:D493-D496, 2004
13. Wagner A, Babb L, Lopez J, et al: ga4gh/vr-spec: 1.0 GA4GH Approved. 2019. https://zenodo.org/record/3572974