Application Notes

GENETEX—a GENomics Report TEXt mining R package and Shiny application designed to capture real-world clinico-genomic data

David M. Miller 1,2 and Sophia Z. Shalhout 1,2

1Division of Hematology/Oncology, Department of Medicine, Massachusetts General Hospital, Boston, Massachusetts, USA and 2Department of Dermatology, Massachusetts General Hospital, Boston, Massachusetts, USA

Corresponding Author: David M. Miller, MD, PhD, Division of Hematology/Oncology, Department of Medicine, Massachusetts General Hospital, 15 Parkman Street, Room 132, Boston, MA 02114, USA; dmiller4@mgh.harvard.edu

Received 12 July 2021; Revised 17 August 2021; Editorial Decision 2 September 2021; Accepted 9 September 2021

ABSTRACT

Objectives: Clinico-genomic data (CGD) acquired through routine clinical practice has the potential to improve our understanding of clinical oncology. However, these data often reside in heterogeneous and semistructured data, resulting in prolonged time-to-analyses.

Materials and Methods: We created GENETEX: an R package and Shiny application for text mining genomic reports from electronic health record (EHR) and direct import into Research Electronic Data Capture (REDCap).

Results: GENETEX facilitates the abstraction of CGD from EHR and streamlines the capture of structured data into REDCap. Its functions include natural language processing of key genomic information, transformation of semistructured data into structured data, and importation into REDCap. When evaluated with manual abstraction, GENETEX had >99% agreement and captured CGD in approximately one-fifth the time.

Conclusions: GENETEX is freely available under the Massachusetts Institute of Technology license and can be obtained from GitHub (https://github.com/TheMillerLab/genetex). GENETEX is executed in R and deployed as a Shiny application for non-R users. It produces high-fidelity abstraction of CGD in a fraction of the time.

Key words: clinico-genomics, data abstraction, electronic health records, Shiny app, REDCap, clinical informatics

INTRODUCTION

Advances in clinical oncology require a deep understanding of cancer biology. Clinico-genomic data (CGD) obtained from routine clinical practice can greatly increase our comprehension of tumor biology. However, there are a number of barriers that impede capitalization of these critical real-world data (RWD). Paramount amongst these obstacles are prolonged time-to-analyses secondary to the difficulties of capturing data from heterogeneous sources, as well as the challenges of processing vast amounts of genomic information. These hurdles increase time-to-insight from RWD and threaten our ability to fully maximize on advances in molecular and information technologies.
We previously published an overview of a methodology and design of a Research Electronic Data Capture (REDCap)-based system to facilitate capture of RWD. REDCap is a web-based EDC utilized by researchers to collect structured data. That platform incorporates a form entitled Genomics Instrument, which provides a structured format for the collection of CGD. This instrument is freely available and can be incorporated into any existing REDCap project.

Here, we present GENETEX (pronounced “genetics”), an R package with a Shiny application front-end, which facilitates the abstraction of CGD from electronic health record (EHR) and streamlines capture of structured data into the Genomics Instrument in REDCap. Its functions include natural language processing of key genomic information, transformation of semistructured data into structured data, and importation into REDCap (Figure 1). GENETEX is executed in R but is deployed as a Shiny application to enhance the user interface for non-R users.

MATERIALS AND METHODS

Software dependencies

GENETEX is written in R (version 4.0.0), organized using roxygen2 and utilizes the following packages dplyr, tidyr, stringr, purrr, REDCapR, magrittr, splitstackshape, and Shiny. For full details, instructions, and examples refer to either our README (https://github.com/TheMillerLab/genetex/blob/main/README.md) or video demonstration (https://github.com/TheMillerLab/genetex/blob/main/Demo_Video.md), both of which can be viewed on the package GitHub page.

Clinical informatics dependencies

GENETEX facilitates the abstraction of medical records for importation into the Genomics Instrument in REDCap. The data dictionary for this form has been previously published.

Comparison of GENETEX to manual abstraction

Sample genomic reports were either generated or obtained from commercial vendors. These data were devoid of protected health information; thus, no IRB was required for this project. Two highly trained abstractors manually abstracted the reports and recorded the time-to-capture for each report. GENETEX was used to abstract these same reports. To simulate the real-world experience, both techniques incorporated a manual visual quality-control step to verify if imported results were accurate. The time spent on this step was included in the total time-to-capture. Agreement rates were compared using R. A paired Wilcoxon test was used to compare the time of manual abstraction with the time to capture CGD with the GENETEX package.

RESULTS

Inputs/user interface

CGD in the real-world setting is predominantly contained in either portable document format documents sent to providers by commercial vendors or in text files contained within EHR. Thus, to facilitate abstraction of these data we developed a browser-based user interface that incorporates text data captured on a clipboard as input in a Shiny application. Text is copied to a computer’s clipboard and pasted into the text area input in the Shiny application (Figure 2).

Users then control additional inputs including the free text of the subject’s record id (required field), REDCap instrument instance (required), lesion tag descriptor (optional field), and date the tissue was obtained (optional). Drop-down inputs are also presented to the user including the selection of the platform used to generate the genomics report (required) and the type of lesion the genomics report was generated from (eg, primary lesion vs metastases; optional). Finally, to direct the data to REDCap, users enter strings of the web address of the REDCap platform (required) as well as the REDCap API Token (required). These inputs are then called to the function genetex_to_redcap by the action button “Run GENETEX to REDCap.”

Server-side functions

The server side of the Shiny application contains the executable code of GENETEX. The package contains a set of functions that then parse, text mine, and transform the input into structured data to serve as the substrate for import into REDCap (Figure 1). Table 1 summarizes the functions and their respective function to extract key elements from the genomics report. We have methods to automatically mine HUGO Gene Nomenclature Committee-approved gene names and detected amino acid and/or nucleotide alterations, tumor mutational burden (tmb), mismatch repair status (mmr), copy number variants (cnv), and mutational signatures. In addition, our implementation transforms the data and links the appropriate CGD with the variables used in the Genomics Instrument so that they can be uploaded into REDCap.

Text mining CGD

Overview of data processing

Following the initial step of securing CGD into the Shiny application, GENETEX converts these character strings to a data frame for text
mining. At this time, CGD from the following platforms are able to be processed by GENETEX: Guardant360,14 FoundationOne,15 Massachusetts General Hospital (MGH) SNaPshot,16 and Brigham and Women’s Hospital (BWH) Oncopanel.17

Due to idiosyncratic differences between these reports, we developed platform-specific functions to text mine data. For example, `gene.variant.isolate.oncopanel()` and `gene.variants.isolate.snapshot()` isolate gene variant data from BWH Oncopanel and MGH SNaPshot reports, respectively. However, in general, following securing CGD, the data are tokenized using the `cSplit()` function from the `splitstackshape` package.

Isolating genomic data with regular expressions
To perform text mining of CGD, we created a number of regular expressions (regex) to identify gene names, nucleotide and amino

---

**Figure 1.** Schema of GENETEX. The GENETEX package takes CGD, which is typically stored in semi-structured data, as an input via a Shiny application user interface. Once the input data have been captured, the package executes a series of server-side functions that text mine CGD reports for relevant genomic data. These structured data are then imported directly into the REDCap electronic data capture system (EDC), placing the data in the Genomics Instrument in REDCap. CGD: clinico-genomic data; REDCap: Research Electronic Data Capture.

**Figure 2.** Browser-based user interface. Depicted is the user interface (UI) of the Shiny app of GENETEX. This UI is produced by running the code in the R script “GENETEX Shiny app. R,” which can be found on GITHUB (https://github.com/TheMillerLab/genetex/blob/main/GENETEX%20Shiny%20app.R).
Table 1. GENETEX functions

| Function                        | Functionality                                                                 |
|---------------------------------|-------------------------------------------------------------------------------|
| genetex_to_redcap()             | Integrates key verbs to provide NLP tools to abstract data from a variety of genomic reports and import them to REDCap |
| gene.variants()                 | Integrates various platform-specific NLP functions to text mine gene names and nucleotide variants from genomic reports and transforms them to structured data for import into REDCap |
| cvn()                           | Integrates various platform-specific NLP functions to text mine gene names and copy number variants data from a variety of genomic reports and transforms them to structured data for import into REDCap |
| mmr()                           | Text mines mismatch repair status from genomic reports and transform it to structured data for import into REDCap |
| mutational.signatures()         | Text mines mutational signatures data from a variety of genomic reports and transforms it to structured data for import into REDCap |
| platform()                      | Applies regular expressions to assign a numerical value for the various platforms used for genomic reports that aligns with the “genomics_platform” field in the REDCap Genomics Instrument |
| genes_regex()                   | Produces a regular expression of over 900 HGNC gene names                      |
| genes_boundary_regex()          | Produces a regular expression of over 900 HGNC gene names as a unique string with word boundaries |
| genomics.tissue.type()          | Applies regular expressions to assign a numerical value for the various platforms used for genomic reports that aligns with the “genomics_platform” field in the REDCap Genomics Instrument |

Notes: Key functions unique to GENETEX with a brief description of action are shown. Description of other functions can be found in the package’s Help Page.

REDCap: Research Electronic Data Capture; NLP: Natural Language Processing.

acid sequences, and cell-free DNA (cfDNA) data within genomic reports. For example, the function genes_boundary_regex() generates a regular expression of >900 gene names surrounded by word boundaries (Figure 3A). We further designed regular expressions to detect nucleotide and amino acid sequences and the magnitude of cfDNA (Figure 3B). The regular expressions were effectively combined to identify only those genomic information of interest (Figure 3C). Finally, we filter out unnecessary elements (eg, the strings “RESULTS”: “Single” “nucleotide” “variants”) that are not intended to be captured in the Genomics instrument (Figure 3D).

Group correlated text
In order to group correlated genomic information (eg, gene name with the associated nucleotide/amino acid variant), GENETEX utilizes keyword-group pairing. A logical vector “keywords” is created using the dplyr function mutate() paired with str_detect() and the regular expression “genes_boundary_regex.” This logical vector becomes the object of the cumsum() function to create the numeric vector “group”; effectively grouping each unique gene name with its correlated data (Figure 3E).

Mapping REDCap variables
In addition to isolating key genomic data from reports, the above regular expressions are also used to map variables used in the Genomics Instrument. The instrument uses the following variable prefixes “variant_gene,” “variant_nucleotide,” “variant_protein,” and “variant_gene_perc_cfdna” to enable tidy data for gene names, nucleotide variants, amino acid variants, and percent cfDNA, respectively. Using the following combination of ifelse() statements, str_detect(), and the aforementioned regular expressions, these variables can be linked to the corresponding data (Figure 3F).

As previously described, in order to produce unique variables that can be linked with other related information, the Genomics Instrument utilizes a numeric suffix linker system (NSLS). The NSLS links related elements of CGD with a character string in the variable name (eg, “variant”) with an underscore and a numeric (e.g. “_1”). Therefore, a given gene will be grouped with its correlated nucleotide/amino acid variants and/or cfDNA information with a unique character string and numeric. For example, the variable prefixes “variant_gene,” “variant_nucleotide,” “variant_protein,” and “variant_gene_perc_cfdna” will all be linked with the same numeric suffix (eg, “_1”). Consequently, these elements can be grouped during analysis with the unique pairing of “variant” and “_1.” Therefore, a final step in creating this unique variable-linked system involves pasting the “var” vector with the “group” vector (Figure 3G).

An analogous approach is used to identify and abstract data on cnvs, tmb, mmr, and mutational signatures. Please see the description file and annotated R scripts contained within the package in GitHub for further details.

Package outputs
The front-end Shiny application is executed with an action button that produces 2 easy-to-view outputs. The first, which can be viewed by clicking on “Report” in the sidebar, produces a verbatimTextOutput of the genomic report to ensure that the correct report was pasted into the textAreaInput. The second one, viewed by clicking on “Data” in the sidebar, is a table of the data executed by genetex_to_redcap() (Supplementary Table S1). This is intended to provide the user with an output of the data generated by the function so that a quality-control step can take place.

Import to REDCap
The data in Table 2 are imported to REDCap from the Shiny application by calling the function readcap_write_oneshot() from the REDCapR package. An example of a portion of that form with data imported from GENETEX is seen in Supplementary Figure S1.

Real-world deployment
In order to evaluate the performance of augmented abstraction with GENETEX in the real-world setting compared to manual capture, we selected 7 genomic reports at random (3 Guardant, 2 Foundation
Medicine, 1 MGH SNaPshot, and 1 BWH Oncopanel) for abstraction. Each report was abstracted independently by 2 data abstractors via manual abstraction, as well as with GENETEX. In total, 744 data elements were captured from these 7 reports. Agreement rates between the 2 human abstractors were 99.19%. Importantly, >99% agreement was reached between both human abstractors and GENETEX (99.33% between abstractor one and GENETEX and 99.19% between abstractor 2 and GENETEX). Given that the agreement between classical and augmented abstraction was high, we next evaluated if the GENETEX pipeline would improve time-to-analysis. The mean time for manual abstraction for each report was 784.5 (range: 220.5–3096.5) seconds compared with 136 (range: 75–216) seconds for augmented abstraction (Wilcoxon test P value = .015625). For augmented abstraction, the average time for capture and processing by GENETEX was 42 s (range: 34–54) with 94 s (range: 40–170) used for abstractor verification.

Limitations and solutions
GENETEX is to be used in conjunction with Genomics Instrument and thus, it is dependent on that form being installed into an REDCap project. However, we have made the data dictionary freely available so that others may incorporate it into their individual project. Importantly, our regular expressions system of mapping variable

Figure 3. (A) Regular expression of gene names. Depicted is a portion of the character vector output of the function “genes_boundary_regex().” This function produces a regular expression that is used by GENETEX to identify gene names in CGD reports. (B) Regular expression of nucleotide and amino acid sequences, and cfDNA. Shown are the 3 regular expressions used to identify nucleotide sequences (“nuc_regex”), amino acid sequences (“aa_regex”), and cfDNA (“cfdna_regex”) contained within CGD reports. (C) Tokenized genomics report. Depicted is a portion of a CGD report that has been tokenized. Here, each word of the report is partitioned into a single cell of the vector “X.” (D) Example of report filtered with genes_nuc_aa_cfdna_regex. Shown is vector “X” from (C), which has been filtered by a regular expression that selects only cells with elements relevant to gene names, nucleotide, and amino acid sequences and cfDNA. The code used for this step is demonstrated above the output. (E) Example of report grouped by gene name. Related tokens are grouped using the “stringr::str_detect()” function by incorporating the regular expression “genes_boundary_regex.” With this method, HUGO gene names serve as the “keyword” and thus the boundary for each group. As a result, the appropriate nucleotide, amino acid, and cfDNA data are linked with the corresponding gene name. (F) Mapping REDCap variables to data elements. Each tokenized data element in vector “X” must be linked with an appropriate variable name from the Genomics Instrument. In this step, the 4 relevant variable stems, “variant_gene,” “variant_nucleotide,” “variant_protein,” and “variant_gene_perc_cfdna” are matched with the relevant data in vector “X” by combining the “ifelse()” and “str_detect()” functions with the regular expressions “genes_boundary_regex,” “nuc_regex,” “aa_regex,” and “cfdna_regex.” (G) Complete mapping REDCap variables to data elements with the NSLS. Each data element in vector “X” must correspond to a unique variable name to be imported into REDCap. Therefore, in this final step, the variable stems “variant_gene,” “variant_nucleotide,” “variant_protein,” and “variant_gene_perc_cfdna” are matched with the relevant data in vector “X” by combining the “ifelse()” and “str_detect()” functions with the regular expressions “genes_boundary_regex,” “nuc_regex,” “aa_regex,” and “cfdna_regex.”
names to key genomic data provides a high degree of flexibility to map alternate variable names for REDCap instruments with different data dictionaries. Additional limitations include the fact that at this time, GENETEX does not support all potential platforms available for CGD. However, due to its open-source position, external developers can perform pull-requests on GitHub for incorporation of additional platforms and future refinement.

CONCLUSIONS
GENETEX is a browser-based application for natural language processing of CGD obtained in routine clinical practice. It facilitates extraction of data from EHR, transformation of semistructured data into a structured format, and loading into REDCap. Its Shiny extension enables non-R users to execute the package without the familiarity of R. Real-world deployment of the GENETEX demonstrated excellent agreement with classical abstraction in roughly 1/5 of the time. Thus, augmented abstraction with browser-based applications can decrease the barrier to data capture and importantly improve time-to-analysis of CGD.

FUNDING
The Harvard Cancer Center Merkel Cell Carcinoma patient registry is supported by grants from Project Data Sphere, the American Skin Association, and ECOG-ACRIN.

AUTHOR CONTRIBUTIONS
DMM created and developed the GENETEX package, authored the manuscript, and granted final approval of the manuscript. SZS contributed to the development of the GENETEX package, including code writing, participated in the authorship of the manuscript, and granted final approval of the manuscript.

SUPPLEMENTARY MATERIAL
Supplementary material is available at JAMIA Open online.

ACKNOWLEDGMENTS
The authors would like to acknowledge Ravikumar Komandur, PhD, Project Director at Project Data Sphere for review and critique of the manuscript; Guardant Health and Foundation Medicine for making sample reports available for the development of this package. GENETEX is for research purposes only. No clinical decisions should be made with the information obtained from its output. This article reflects the views and work (including development and use of GENETEX) of the authors and should not be construed to represent the work, policies, of any of the vendors whose reports were used to develop GENETEX and whose reports may be provided as part of the package.

CONFLICT OF INTEREST STATEMENT
None declared.

DATA AVAILABILITY
The data/code for this application can be found in our GITHUB repository (https://github.com/TheMillerLab/genetex). Data available from the Dryad Digital Repository: https://doi.org/10.5061/dryad.dbrv15f25.

REFERENCES
1. Shalhout S, Miller D. Optimizing Real-World Data Collection: Clinical Genomics. 2020. https://www.themillerlab.io/post/optimizing_rwd_collection-clinical_genomics/ Accessed June 1, 2021.
2. Harris PA, Taylor R, Thielke R, et al. Research electronic data capture (redcap)–a metadata-driven methodology and workflow process for providing translational research informatics support. J Biomed Inf 2009; 42: 377–81.
3. Miller D, Shalhout S. Optimizing Real-World Data Collection: Genomics Electronic Data Capture Instrument. 2021. https://www.themillerlab.io/post/optimizing_rwd_collection_genomics_instrument/ Accessed June 1, 2021.
4. Wickham H, Danenberg P, Eugster M. Roxygen2: In-source documentation for R (version 6.0.13). R package Version 1.0.1. 2013. https://cran.r-project.org/package=roxygen2 Accessed June 1, 2021.
5. Wickham H, Francois R, Henry L, Muller K. Dplyr: a grammar of data manipulation. R Package Version 1.0.5. 2021.
6. Wickham H. Tidyverse: tidy data. R Package Version 1.3.3. 2013.
7. Wickham H, Hester J, Francois R. Rcr: read rectangular text data. R Package Version 1.4.0. 2020.
8. Wickham H. Stringr: simple, consistent wrappers for common string data. R Package Version 1.0.5. 2021.
9. Henry L, Wickham H. Purrr: functional programming tools. R Package Version 0.3.4. 2020.
10. Beasley W, Bard D, Wilson T, et al. REDCapR: interaction between r and redcap. R Package Version 0.11.0. 2020.
11. Bache S, Wickham H, Henry L. Magrittr: a forward-pipe operator for R. R Package Version 2.0.1. 2020.
12. Mobility A. Splintaskshape: stack and reshape datasets after splitting concatenated values. R Package Version 1.4.8. 2019.
13. Chang W, Cheng J, Allaire J, Xie Y, McPherson J. Shiny: web application framework for r. R Package Version 1.6.0. 2018.
14. GuardantHealth. https://guardanthealth.com Accessed June 1, 2021.
15. Foundation medicine. https://www.foundationmedicine.com Accessed June 1, 2021.
16. Massachusetts General Hospital Center for Integrated Diagnostics. https://www.massgeneral.org/pathology/services/center-for-integrated-diagnostics Accessed June 1, 2021.
17. Brigham and Women’s Hospital Center for Advanced Molecular Diagnostics. https://www.brighamandwomens.org/pathology/center-for-advanced-molecular-diagnostics/molecular-diagnostics-laboratory Accessed June 1, 2021.