Informatics in Biology, Health, and Food

Interdisciplinary Explorations in Life Science

Volume 3. Number 1. Jun. 2026 DOI: 10.22034/ibhf.2026.582464.1053

Original Article

TCGA-DL: A Lightweight Windows Application for Code-Free TCGA Data Download

Amirmohammad Asgari[1], Mohammadreza Shahbazi[1], Abolfazl Ghasemi[1], Mahan Mirzade[1], Siamak Salimy[1]*

[1] Department of Computer Engineering, National University of Skills (NUS), Tehran, Iran

Received: 2026/05/25 **Revised:** 2026/06/20 **Accepted:** 2026/07/07

Abstract

For many biologists, accessing The Cancer Genome Atlas (TCGA) data can be technically demanding. For example, a wet-lab scientist looking to retrieve gene expression data for breast cancer samples must locate metadata among files, then craft a manifest and execute commands often resulting in errors, incomplete downloads, or wasted hours. TCGA is a pivotal resource for cancer genomics, yet its utility is limited for non-programmers due to reliance on command-line tools or complex scripting for data download. Although the NCI's GDC Data Transfer Tool supports manifest-based downloads, it lacks a graphical interface and offers no interactive data exploration. This study aimed to introduce and evaluate TCGA-DL as a Windows-based graphical application for retrieval and organization of open-access TCGA data using GDC-compatible manifest files. The application provides graphical data selection combined with GDC-compatible, manifest-driven downloads from the GDC portal. Users can browse and filter datasets by cancer type or project, data category, and sample attributes via a point-and-click interface. For RNA-seq gene expression quantification files, the expanded benchmark used small, medium, and large GDC-compatible manifests (10, 100, and 300 files; approximately 42.34 MB, 423.39 MB, and 1.27 GB) with 10 repeated runs per tool. TCGA-DL completed the three datasets in 47.01 ± 3.80 s, 452.04 ± 5.57 s, and 1301.39 ± 66.73 s, respectively, showing lower mean completion times than the GDC Data Transfer Tool in this specific benchmark. TCGAbiolinks showed higher raw download speed in the medium and large datasets, but required installation of R, Bioconductor dependencies, and script-based execution, whereas TCGA-DL provided a Windows GUI workflow for users who do not work in R. TCGA-DL may help reduce usability barriers for biologists and clinical researchers who need access to TCGA data without extensive command-line interaction. It is a Windows-based tool (Python 3 and Tkinter) distributed under the MIT license.

Keywords: GDC; graphical user interface; cancer bioinformatics; manifest-based retrieval; open-access genomics

Cite this article: TCGA-DL: A Lightweight Windows Application for Code-Free TCGA Data Download. *Informatics in Biology, Health, and Food*. 2026;3(1):50-62.

Corresponding authors: Siamak Salimy **Email**: salimy@ut.ac.ir

Introduction

The Cancer Genome Atlas (TCGA) represents one of the most comprehensive public repositories of multi-omics and clinical data across cancer types, enabling breakthroughs in precision oncology, biomarker discovery, and systems biology (1,2,4,5). Recent TCGA- and multi-omics-based cancer studies further demonstrate how public cancer datasets can support survival modeling, biomarker discovery, and precision-oncology workflows (24-29). Related network-based omics studies have also used gene co-expression and metagenomic co-occurrence analyses to identify biomarkers and community alterations in disease contexts (32,33). To facilitate data access, the National Cancer Institute (NCI) provides the Genomic Data Commons (GDC) portal, which hosts TCGA datasets alongside a programmatic API and a command-line data transfer tool (3,6). While powerful, these official interfaces require users to navigate complex metadata structures, manually construct manifest files, and manage downloads via terminal commands tasks that pose significant barriers for bench biologists, clinicians, and researchers without programming experience.

Several community-developed tools have attempted to lower this barrier, including TCGA-focused acquisition, processing, and visualization resources (7-10,12,15,16,18,19). For instance, TCGAbiolinks (an R/Bioconductor package) offers rich querying and download capabilities but remains confined to the R ecosystem and demands coding proficiency (12). Web-based platforms such as cBioPortal (13) and UCSC Xena (14) provide user-friendly exploration, but either limit direct bulk data export or lack support for raw sequencing files. More recent online or package-based resources, including TCGAnalyzeR, TCGAplot, and PANDA, further demonstrate the growth of analysis- and visualization-centered TCGA tools rather than native Windows downloaders (15,16,19). The official GDC Data Transfer Tool supports manifest-based downloads, an efficient and standardized method aligned with the GDC data-access ecosystem (3,6,17). Still, its command-line workflow may be less accessible to non-programmers because it provides no graphical interactivity, visual feedback, or intuitive filtering. TCGA-DL combines bulk manifest-based downloads of raw and processed TCGA data with a graphical Windows interface, providing a complementary option to coding-dependent or exploration-focused tools. This integrated approach is intended to help non-programmers access selected TCGA datasets through a simplified workflow.

To overcome these limitations, TCGA-DL was developed as a Windows desktop application using Python and the Tkinter framework (30), providing a graphical interface for TCGA data retrieval. TCGA-DL combines an intuitive file-based manifest loader with a download engine that accesses data directly from the GDC. The user interface layout was guided by established human-computer interaction (HCI) principles, particularly Nielsen's usability heuristics (31). For instance, the design emphasizes visibility of system status through real-time progress bars and clear completion messages and incorporates error prevention by disabling incompatible actions and providing informative error prompts if a user selects an invalid file or encounters a network issue. These choices are intended to help biologists and clinicians without a programming background use the application more confidently and effectively. Multi-threading support is planned for a future release. Users may select one or multiple manifest files exported from the GDC portal, specify a destination directory, and download all files through a graphical interface with real-time progress tracking. The application reads GDC-compatible manifests and performs manifest-driven downloads using HTTP streaming. In the expanded benchmark reported here, three manifest sizes were evaluated: small (10 files, 42.34 MB), medium (100 files, 423.39 MB), and large (300 files, 1.27 GB), each repeated 10 times per tool. These results should be interpreted as benchmark evidence for the tested GDC-hosted open-access RNA-seq gene expression quantification files rather than as a universal claim for all TCGA-scale data types or network environments. By combining a graphical interface, manifest handling, and

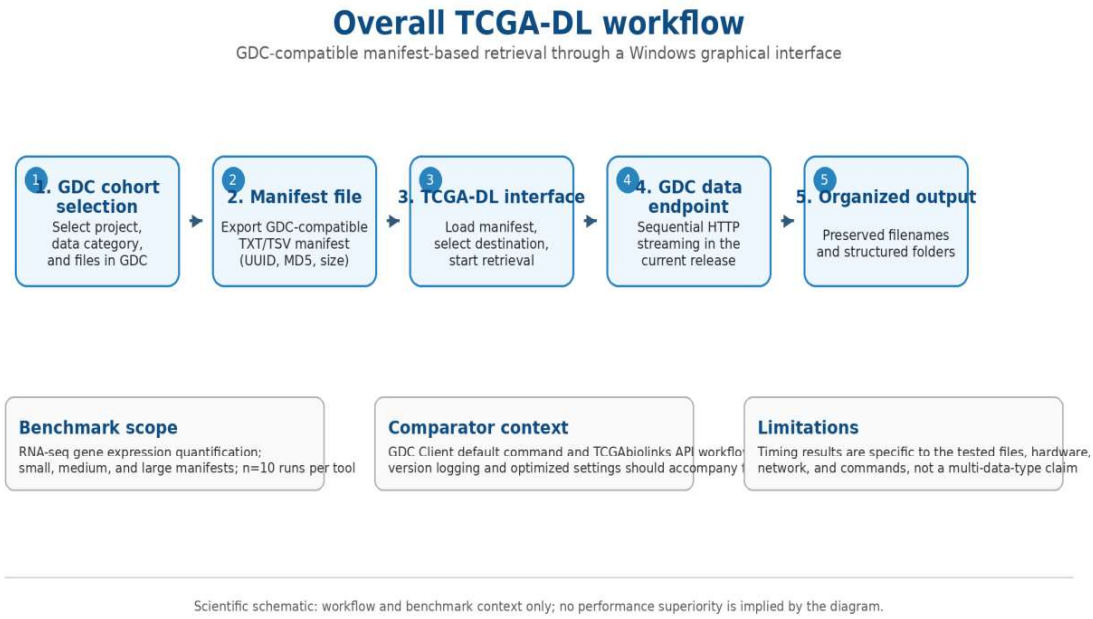

Figure 1. Overall TCGA-DL workflow from GDC cohort selection to organized analysis-ready output folders.

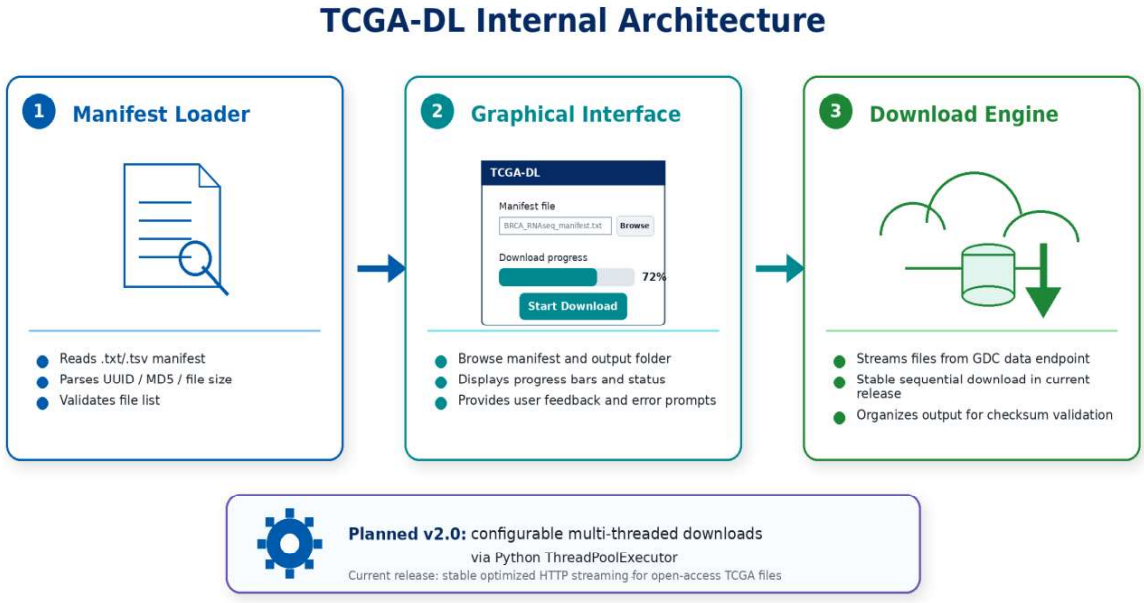

Figure 2. Internal architecture of TCGA-DL showing the Manifest Loader, Graphical Interface, Download Engine, and planned multithreaded extension.

automatic organization, TCGA-DL provides a practical GUI-based solution for accessing TCGA-compatible datasets on Windows systems. Developed in Python 3.9+ with the Tkinter framework for the graphical interface and the requests module for HTTP communication, the current version can be used through a packaged Windows executable or run from source.

Architecture Overview

To provide clarity and guide readers, TCGA-DL is organized into three core components: the Manifest Loader (for preparing download files), the Graphical Interface (for user interaction and progress tracking), and the Download Engine (for handling the data transfer process). The application follows a modular design consisting of these three core components:

Manifest Loader: Reads one or multiple GDC manifest files (.txt or .tsv) selected by the user and prepares them for download.

Graphical Interface: Provides simple buttons and progress bars built with Tkinter for browsing manifest files, selecting output folders, and tracking download progress.

Download Engine: Reads a GDC-compatible manifest file selected by the user and performs sequential HTTP streaming downloads through the GDC data endpoint. Configurable multi-threading via ThreadPoolExecutor is not used in the current release and is planned for the next release.

The overall workflow emphasizes how TCGA-DL converts a GDC cohort selection into a manifest-driven download process and then organizes the retrieved files into analysis-ready folders (Figure 1).

Manifest-Based Download Workflow

To maintain compatibility with the official GDC infrastructure, TCGA-DL fully supports the manifest file format used by the GDC portal (UUID, MD5 checksum, file size) (3,6,17). After cohort selection, the tool:

1. Loads the manifest file exported from the GDC Data Portal.
2. Parses UUIDs and prepares GDC data endpoint requests.
3. Downloads files through https://api.gdc.cancer.gov/data/ using sequential HTTP streaming in the current release; configurable concurrent downloads are planned for v2.0.

Performance Optimization

The current implementation uses sequential HTTP streaming with large chunk sizes to reduce local I/O overhead. No concurrent or multi-threaded downloading is enabled in the current release. The current release is distributed as a Windows application; the next release will add configurable multi-threading via Python ThreadPoolExecutor to improve throughput on multi-core systems and high-bandwidth networks (23).

Performance evaluation uses metrics that are standard for Windows application and data-transfer assessment: total download time, average throughput, peak CPU usage, peak memory consumption, disk I/O latency, completion status/success rate, and scalability with manifest size. CPU, memory, and disk counters are included because Windows Performance Counters are designed to collect processors, memory, and disk usage statistics for diagnosing software bottlenecks (22). Download time and throughput quantify end-user acquisition performance, while success rate and checksum validation reflect transfer integrity checks against GDC manifest metadata and official data-transfer behavior (3,6,17,21). The performance benchmark was expanded from a single small manifest into a repeated three-size evaluation using small, medium, and large GDC-compatible manifests (10, 100, and 300 files). TCGA-DL, the official GDC Data Transfer Tool, and TCGAbiolinks were compared under the same hardware, output-directory clearing procedure, and network environment. TCGA-DL was executed through its graphical workflow by selecting the manifest and output folder and initiating the download. The GDC command-line workflow used GDC Data Transfer Tool version 2.3 with the standard command `gdc-client download -m <manifest_file> -d <output_directory>` without additional optimization flags. No optimized GDC settings (e.g., alternative connection counts, chunk sizes, checksum-disabling options, or retry-parameter tuning) were evaluated, so the GDC comparison should be interpreted as a default-command comparison rather than a fully optimized transfer benchmark. TCGAbiolinks was executed in R using `GDCdownload(query, method = "api", directory = <output_directory>, files.per.chunk = 10) `. Each dataset-tool combination was repeated 10 times, and results are reported as mean ± SD. The benchmark expansion refers to manifest size, file count, and repeated timing for RNA-seq gene expression quantification files; it should not be interpreted as a complete

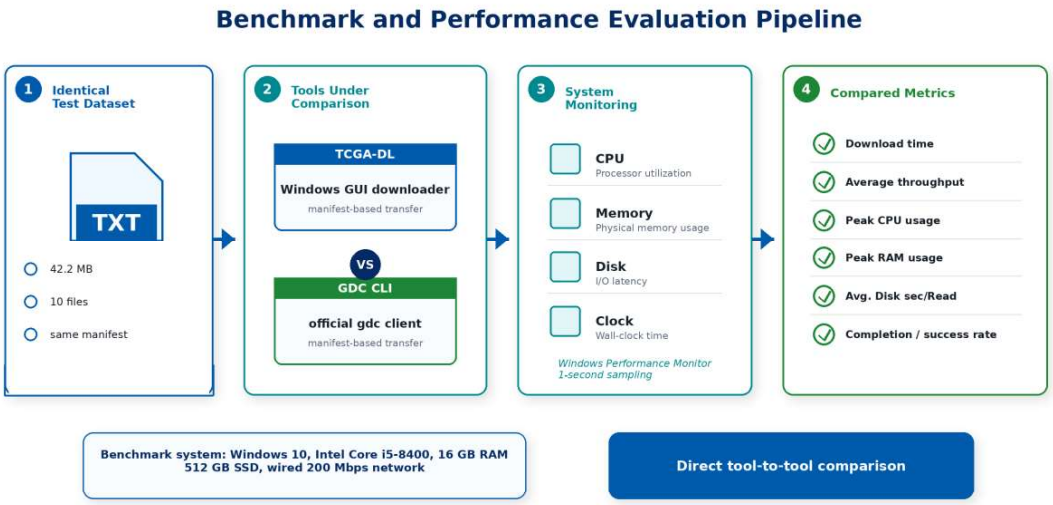

Figure 3. Benchmark and performance evaluation pipeline. The same general workflow was applied to small, medium, and large GDC-compatible manifests, while resource metrics and completion status were captured during the download process. The schematic illustrates the benchmarking workflow; detailed expanded timing results are reported in Table 1 and Figure 5.

multi-data-type benchmark.

The GUI is designed for interaction through standard desktop controls. All functions, from manifest selection to download initiation, are performed through a button-driven CustomTkinter interface with standard interface controls and optional dark/light display modes. Contextual tooltips, error messages (such as signaling invalid filters or network issues), and auto-saved session states further enhance usability. When a download fails mid-stream, the interface provides clear, actionable feedback to the user by displaying a descriptive error message and options to retry, skip the affected file, or resume from the point of interruption where possible. This design philosophy emphasizes graceful degradation so that unexpected issues do not result in silent failure or data loss, but instead offer users guidance on how to proceed and maintain progress. The application is distributed as a Python package and as a Windows executable (built with PyInstaller), so users of the executable do not need to configure a Python environment or additional bioinformatics libraries.

The source code, documentation, example workflows, and a short demonstration video are publicly available on GitHub under the MIT license at: https://siamak-salimy.github.io/HMNexus/.

Results

To evaluate the resource profile and download performance of TCGA-DL, system resource usage was monitored using Windows Performance Monitor with a 1-second sampling interval, and completion time was recorded across expanded benchmark datasets. The download-time benchmark used GDC-compatible open-access RNA-seq gene expression quantification manifests with 10, 100, and 300 files. The medium and large manifests contained files from HCMI-CMDC and CPTAC-3, while the small manifests contained HCMI-CMDC files only. All tests were conducted on a desktop PC running Windows 10 (64-bit), equipped with an Intel Core i5-8400 CPU (6 cores, 2.8 GHz), 16 GB RAM, and a 512 GB SSD, over a wired Ethernet connection with a measured download bandwidth of 200 Mbps. Output folders were cleared before each run, and completion was verified by counting the downloaded files. The benchmark focused on timing and successful retrieval of gene expression quantification files; systematic timing tests across clinical, methylation, copy- number, and other GDC data

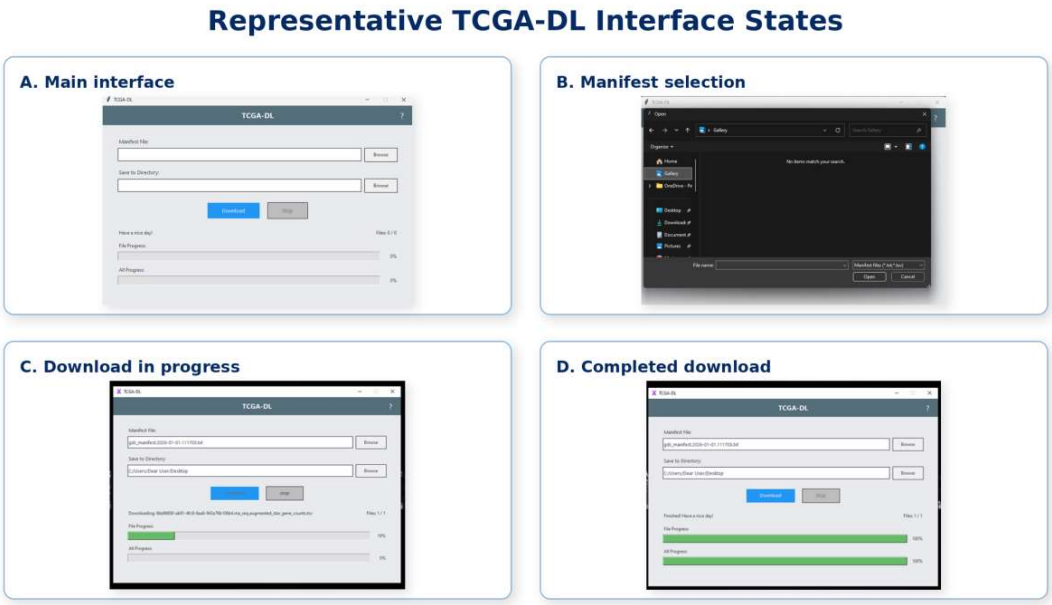

Figure 4. Representative TCGA-DL graphical interface states: (A) main interface, (B) manifest-file selection, (C) download in progress, and (D) completed download status.

types were not performed in this study and are identified as future validation targets.

In the local resource-monitoring subset, TCGA-DL showed low resource usage: peak CPU usage ranged from 6.2% to 6.8% (mean 6.5%, SD 0.2%), while peak RAM consumption ranged from 150 MB to 158 MB (mean 154 MB, SD 3.4 MB). Disk I/O activity remained low throughout (average Disk sec/Read < 0.005 s). The application completed the download while displaying a graphical interface with real-time progress and a final "Finished! Have a nice day!" status (Figure 4).

The official GDC Data Transfer Tool (CLI) was treated as the primary official comparator because it is a standard GDC-associated transfer utility (3,6,21). TCGAbiolinks was retained as an R/Bioconductor comparator because it provides widely used GDC query and download functions (12). In the expanded benchmark, all three tools successfully retrieved the expected files in all evaluated runs. TCGA-DL and the GDC Client were tested with the same GDC-compatible manifest files, while TCGAbiolinks was matched to the same file UUIDs through GDC query results and executed with `method = "api"`. The GDC Client was evaluated as GDC Data Transfer Tool version 2.3 using the default manifest command and no optimized settings. Therefore, the results are reported as environment-specific timing observations for the stated commands and conditions, not as a fully optimized head-to-head evaluation. Future replication archives should include `gdc-client --version`, `R.version.string`, `packageVersion("TCGAbiolinks")`, and raw run logs.

Overall, the expanded benchmark showed lower mean completion times for TCGA-DL than for the GDC Client across the three tested RNA-seq gene expression quantification datasets. TCGA-DL completed the small, medium, and large datasets in 47.01 ± 3.80 s, 452.04 ± 5.57 s, and 1301.39 ± 66.73 s, respectively, while the GDC Client required 71.49 ± 6.73 s, 650.19 ± 12.26 s, and 2015.18 ± 38.92 s. TCGAbiolinks achieved the shortest mean completion time in the medium and large datasets (375.33 ± 184.86 s and 736.93 ± 254.11 s), indicating strong raw API-based download performance.

No separate manual-per-file timing benchmark was performed. Therefore, the manuscript does not claim that the manifest-driven workflow is independently faster than every possible manual or sequential download procedure. The evidence supports a workflow-level interpretation: a single GDC-compatible manifest defines the complete file set, preserves file identifiers and metadata, reduces repeated manual file selection steps, and allows batch retrieval with a

Table 1. RNA-seq download-time benchmark across small, medium, and large GDC-compatible manifests (n = 10 runs per tool; GDC Data Transfer Tool v2.3 default command; no optimized-setting comparison)

Dataset	Files / size	TCGA-DL	GDC CLI	TCGAbio.	Success
Small	10 files / 42.34 MB	47.01 ± 3.80 s	71.49 ± 6.73 s	66.52 ± 78.97 s	100%
Medium	100 files / 423.39 MB	452.04 ± 5.57 s	650.19 ± 12.26 s	375.33 ± 184.86 s	100%
Large	300 files / 1.27 GB	1301.39 ± 66.73 s	2015.18 ± 38.92 s	736.93 ± 254.11 s	100%

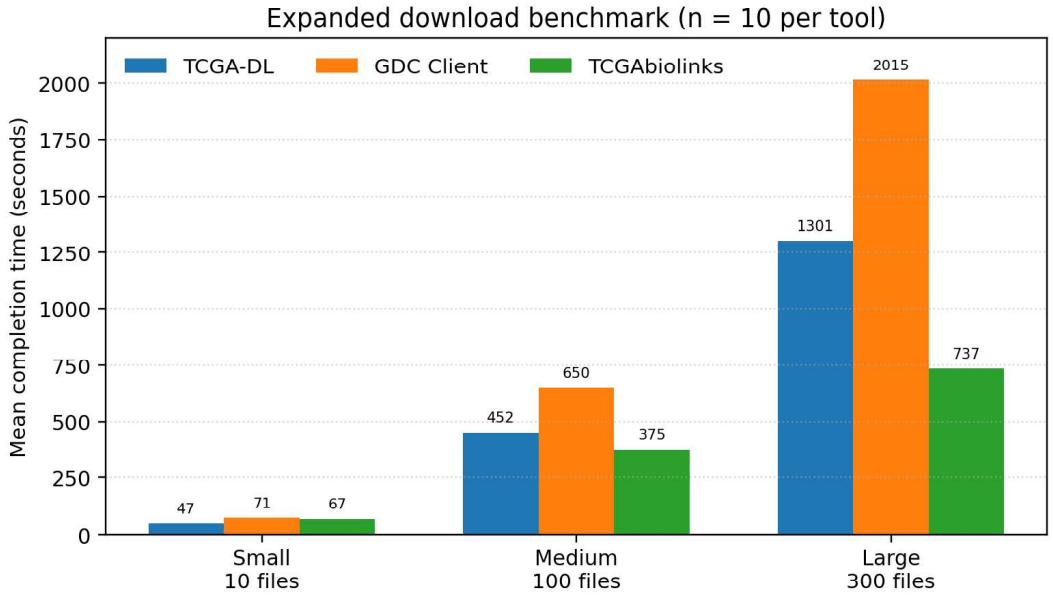

Figure 5. RNA-seq download benchmark results across small, medium, and large manifest sizes. Bars show mean completion time across 10 runs per tool; lower values indicate faster completion. TCGAbiolinks showed the shortest mean time in medium and large datasets, but its use requires R/Bioconductor installation and script execution; TCGA-DL is interpreted here as a Windows GUI alternative focused on usability rather than as the fastest possible transfer engine.

reproducible output structure. A dedicated user-task benchmark comparing manifest-based and manual per-file retrieval would be required to quantify this workflow advantage separately.

User Accessibility and Workflow Integration

For many wet-lab biologists, downloading genomics data from TCGA often involves complex software installations, command-line execution, or manual file management before analysis can begin; this challenge has motivated multiple TCGA access and helper tools (7-10,12). To provide a preliminary usability assessment, five wet-lab biologists without prior command-line experience were recruited by convenience sampling from the local genomics team. Each participant was asked to complete the same BRCA download task with three methods: (1) TCGA-DL, (2) GDC CLI, and (3) TCGAbiolinks. The task consisted of loading or specifying the same manifest/query, selecting an output directory, initiating the download, and locating the downloaded files. Tool order was rotated across participants to reduce order effects. Before each method, participants received only a brief orientation to the interface or command syntax; no hands-on assistance was

provided during task execution. Task time was measured from the start of tool interaction to successful file location, and errors were recorded when installation, configuration, command execution, or file retrieval could not be completed without expert help. After completion, participants rated perceived ease of use using a five-point Likert-style post-task survey. This assessment involved software usability only, did not collect biomedical or personal health data, and was conducted with participants' verbal agreement. All participants completed the task with TCGA-DL in under five minutes. As a standalone Windows executable, TCGA-DL operated without setup or dependencies, resulting in zero errors. In contrast, none of the participants could install or correctly configure GDC CLI or TCGAbiolinks without expert assistance. Participants rated TCGA-DL 4.8 out of 5 for ease of use in the post-task survey.

Downloaded files were automatically organized into a hierarchical folder structure (e.g., BRCA/clinical/, BRCA/rnaseq/), with filenames preserved as provided by GDC. This structure is compatible with downstream tools such as DESeq2, limma, and scikit-learn, reducing the need for manual file sorting. For example, RNA-seq count files can be loaded into DESeq2 with a single R command, enabling differential expression analysis to begin after download. This integration may reduce setup time and lower the risk of file-management errors during data preparation for downstream analyses, including broader multi-omics cancer workflows where reproducible data organization is essential (11).

Robustness and Compatibility

Preliminary compatibility checks were performed using selected GDC-hosted harmonized manifests (3,6,17), including representative open-access data categories such as processed expression, copy number, and clinical data. These checks indicated that TCGA-DL could parse and download files from the tested manifests without application-level crashes in the evaluated scenarios. MD5 checksum comparison was performed where checksum values were available in GDC manifest metadata. Because large-scale robustness testing was not the primary endpoint of the present benchmark, no claim is made here regarding stress-tested handling of more than 10,000 manifests, universal MD5 success across all GDC data types, or failure rates of other tools without supporting logs. These findings should therefore be interpreted as preliminary and require further validation across larger manifests, different network conditions, and additional GDC data categories. Multi-threading support, which may further improve performance for larger manifests, is planned for the next release.

For malformed or outdated manifests, TCGA-DL is described as providing basic workflow-level error handling rather than full validation of all possible GDC states. The application checks whether a selected manifest can be read as a tabular GDC-compatible file and whether required fields such as file identifiers, filenames, and checksum-related metadata are present when available. Rows with missing identifiers, unreachable GDC endpoints, or files no longer available through the current GDC portal are not interpreted as successful downloads; instead, these cases are reported to the user through status messages or failed-transfer logs. Because systematic malformed-manifest stress testing was not performed in the present study, robustness against obsolete or manually edited manifests is presented as a current usability safeguard and an area for further validation rather than as a fully benchmarked claim.

Implications for Cancer Research Laboratories

TCGA-DL addresses two persistent challenges in cancer genomics: the technical barrier to data access and the manual burden associated with command-line data retrieval, both of which remain important considerations in precision-oncology bioinformatics workflows (18,20). By combining a CustomTkinter-based graphical interface with a manifest-driven download engine, the tool enables both user interaction and manifest-based data retrieval. The current release uses sequential downloading; multi-threading is planned for a future release and will be evaluated separately. Several key features of TCGA-DL are designed to align with FAIR-oriented workflow principles: standardized use of GDC manifest files and preserved filenames support Findability; distribution as a Windows

executable with a graphical interface supports access for non-technical researchers; and preservation of original structure with MD5 metadata supports data integrity and Reusability within and beyond the original workflow. This mapping should be interpreted as a design rationale rather than a complete FAIR certification.

TCGA-DL enables researchers without programming expertise to retrieve GDC-compatible datasets through a graphical workflow. In the expanded benchmark, TCGA-DL showed lower mean completion times than the GDC Client across small, medium, and large manifests within the tested RNA-seq gene expression quantification datasets. TCGAbiolinks showed higher raw speed in the larger datasets, but its use requires R installation, Bioconductor package management, and script execution. Thus, the main contribution of TCGA-DL is not a claim of being the fastest possible transfer engine, but rather a balance of performance, accessibility, and graphical usability for non-programming users.

This approach differs in scope from existing solutions. The GDC Data Transfer Tool is a robust official command-line utility that supports manifest-based downloading and resume functionality, and TCGA-DL is not intended to replace it for advanced command-line users (3,6). Instead, TCGA-DL targets non-programming Windows users who need a graphical interface for GDC-compatible open-access downloads. TCGAbiolinks provides extensive analytical features and can achieve high download speed through its API-based workflow, but it requires R proficiency and installation of additional packages (12). Web platforms such as cBioPortal and UCSC Xena excel at interactive exploration but do not support bulk export of raw or processed data in GDC-native formats, which can limit their use for custom downstream analyses (13,14). In summary, TCGA-DL addresses a specific usability gap by providing a manifest-based graphical gateway while preserving compatibility with the official GDC data-access ecosystem.

The reliance on GDC manifest files is a deliberate design choice that ensures interoperability with the official data ecosystem. Users can generate manifests via the GUI or import externally created ones (e.g., from the GDC portal), enabling flexible hybrid workflows (3,6,17). The relationship between TCGA-DL and FAIR principles should be interpreted as a workflow-level alignment rather than a formal FAIRness assessment of the underlying TCGA/GDC dataset. TCGA-DL does not modify or curate TCGA metadata; instead, it preserves GDC-provided file identifiers, manifest structure, filenames, and organization output. In this limited sense, the tool supports findability through manifest identifiers, accessibility through standard GDC download endpoints, interoperability through GDC-compatible manifest formats, and reusability through reproducible folder organization and retained metadata. Accordingly, the FAIR mapping is presented as conceptual and operational support for reproducible data access, not as evidence of full FAIR compliance by the software itself.

TCGA-DL has limitations. It currently supports only open-access TCGA data; integration with controlled-access datasets requiring NIH authentication and dbGaP authorization is planned for a future release (3,6). Additionally, the present benchmark was limited to RNA-seq gene expression quantification files and did not test clinical, methylation, copy-number, imaging, or controlled-access data types. Concurrent downloading can increase speed but may saturate bandwidth on shared networks; users can manage this trade-off by configuring the thread limit in future versions. Finally, the tool focuses exclusively on data retrieval rather than analysis; this narrow scope supports maintainability and usability, consistent with the Unix philosophy of "do one thing well."

The following next steps are proposed to address current limitations:

- Integrate controlled-access TCGA data support (implement NIH dbGaP authentication and authorization workflows)
- Develop and test OAuth-based login modules for secure user authentication
- Enable user configuration of thread limits to optimize bandwidth management and support additional GDC-hosted projects and data types
- Invite community contributions to all

Table 2. Feature comparison of TCGA-DL with existing TCGA data access tools (3,7-10,12-16,18,19)

Feature	TCGA-DL	GDC Data Transfer Tool	TCGAbiolinks	cBioPortal	UCSC Xena
Graphical User Interface (GUI)	Yes	No	No (R console)	Web-based	Web-based
Script-independent operation	Yes	No	No	Yes	Yes
Manifest-based download support	Yes	Yes	Yes (via GDC API)	No	No
Concurrent/multi-threaded download	Planned for v2.0	No	Only with manual setup	Not applicable	Not applicable
Automatic file organization	Yes	No	Partial (user-defined)	No	No
Cross-platform (Windows/macOS/Linux)	Windows only	Yes	Yes (via R)	Web	Web
Bulk export of raw/processed files	Yes	Yes	Yes	Limited	Limited
Open-access TCGA support	Yes	Yes	Yes	Yes	Yes
Controlled-access support	Planned	Yes	Yes	No	No
Primary focus	Data retrieval	Data retrieval	Data retrieval + analysis	Data exploration	Data exploration
License	MIT	Not specified (NCI tool)	GPL-3	BSD/MIT	Public

aforementioned areas, particularly secure access protocols and testing.

These milestones are proposed as future development and validation directions.

Future development plans include extending TCGA-DL to support additional GDC-hosted projects (e.g., TARGET and CPTAC), evaluating controlled-access workflows that require NIH/dbGaP authorization, adding optional sample metadata summary views, and testing configurable concurrent downloading in a separate validated release. These items are presented as planned development directions rather than implemented features of the current release. Community contributions are encouraged via the open-source repository.

To contextualize TCGA-DL within the existing ecosystem of TCGA data access tools, its core features are compared with widely used alternatives in Table 2 (3,7-10,12-16,18,19). In the scope of this comparison, TCGA-DL combines three criteria emphasized in this manuscript: a native graphical user interface, direct manifest-based downloading, and script-independent accessibility. This comparison highlights key differentiators and the specific access gap addressed by the tool.

Conclusion

TCGA-DL provides a graphical approach for downloading TCGA data through a CustomTkinter-based interface, addressing a practical access gap among existing GDC, R-based, and web-based TCGA resources (3,7-10,12-16,18,19). By leveraging manifest-driven streaming downloads, it supports reproducible retrieval workflows while maintaining compatibility with the official GDC data infrastructure. Designed specifically for biologists and clinicians without programming expertise, TCGA-DL may improve practical access to large-scale cancer genomics data and addresses an access gap in current analytical workflows. TCGA-DL is a Windows-based application distributed under the MIT License and freely available at: https://siamak-salimy.github.io/HMNexus/.

Availability of data and materials

TCGA-DL is open-source software released under the MIT License. The complete source code, installation instructions, user documentation, and demonstration videos are publicly available on GitHub at: https://github.com/Siamak-salimy/HMNexus.
All public GDC-hosted data used for benchmarking are freely accessible through the Genomic Data Commons (GDC) at https://portal.gdc.cancer.gov/ (3,6). The expanded timing benchmark used open-access RNA-seq gene expression quantification files from HCMI-CMDC and CPTAC-3 manifests, while additional compatibility examples included representative TCGA/GDC manifests where available. No proprietary or restricted datasets were used in this study.
The current TCGA-DL release runs natively on Windows and is available as a packaged executable for users who prefer not to run the Python source code directly. A Python source distribution is also provided for users who wish to inspect or modify the code.

Acknowledgments
The preparation of this manuscript was supported by artificial intelligence tools. Grammar, spelling, style, punctuation, and overall language polishing were performed using Grammarly Premium with a fully licensed and legally obtained subscription. No artificial intelligence was used for scientific content generation, experimental design, data analysis, benchmarking, or interpretation of results. All scientific decisions and technical content remain the sole responsibility of the authors.

References
1. World Health Organization. Global antimicrobial Weinstein JN, et al. The Cancer Genome Atlas Pan-Cancer analysis project. Nat Genet. 2013;45(10):1113-1120. doi: 10.1038/ng.2764.
2. Liu J, et al. An integrated TCGA pan-cancer clinical data resource to drive high-quality survival outcome analytics. Cell. 2018;173(2):400-416. doi: 10.1016/j.cell.2018.02.052.
3. Genomic Data Commons (GDC). Available at: https://gdc.cancer.gov. Accessed 2026 Jun 8.
4. Tomczak K, Czerwinska P, Wiznerowicz M. The Cancer Genome Atlas (TCGA): an immeasurable source of knowledge. Contemp Oncol (Pozn). 2015;19(1A):A68-A77. doi: 10.5114/wo.2014.47136.
5. Hutter C, Zenklusen JC. The Cancer Genome Atlas: Creating Lasting Value beyond Its Data. Cell. 2018;173(2):283-285. doi: 10.1016/j.cell.2018.03.042.
6. Jensen MA, et al. The NCI Genomic Data Commons as an engine for precision medicine. Blood. 2017;130(4):453-459. doi: 10.1182/blood-2017-03-735654.
7. Chandran UR, et al. TCGA Expedition: A Data Acquisition and Management System for TCGA Data. PLoS One. 2016;11(10):e0165395. doi: 10.1371/journal.pone.0165395.
8. Samur MK. RTCGAToolbox: A New Tool for Exporting TCGA Firehose Data. PLoS One. 2014;9(9):e106397. doi: 10.1371/journal.pone.0106397.
9. Wei L, et al. TCGA-assembler 2: software pipeline for retrieval and processing of TCGA/CPTAC data. Bioinformatics. 2018;34(9):1615-1617. doi: 10.1093/bioinformatics/btx812.
10. Baumann AA, et al. TCGADownloadHelper: simplifying TCGA data extraction and preprocessing. Front Genet. 2025;16:1569290. doi: 10.3389/fgene.2025.1569290.
11. Cantini L, et al. Benchmarking joint multi-omics dimensionality reduction approaches for the study of cancer. Nat Commun. 2021;12(1):124. doi: 10.1038/s41467-020-20430-7.
12. Colaprico A, et al. TCGAbiolinks: an R/Bioconductor package for integrative analysis of TCGA data. Nucleic Acids Res. 2016;44(8):e71. doi: 10.1093/nar/gkv1507.
13. Cerami E, et al. cBioPortal for cancer genomics. Cancer Discov. 2012;2(5):401-404. doi: 10.1158/2159-8290.CD-12-0095.
14. Goldman MJ, et al. Visualizing and interpreting cancer genomics data via the UCSC Xena platform. Nat Biotechnol. 2020;38:675-678. doi: 10.1038/s41587-020-0546-8.
15. Zengin T, Masud BA, Onal-Suzek T. TCGAnalyzeR: An Online Pan-Cancer Tool for Integrative Visualization of Molecular and

Clinical Data of Cancer Patients for Cohort and Associated Gene Discovery. Cancers. 2024;16(2):345. doi: 10.3390/cancers16020345.
16. Liao C, et al. TCGAplot: an R package for integrative pan-cancer analysis and visualization of TCGA multi-omics data. BMC Bioinform. 2023;24:483. doi: 10.1186/s12859-023-05615-3.
17. Gao GF, et al. Before and After: Comparison of Legacy and Harmonized TCGA Genomic Data Commons' Data. Cell Syst. 2019;9(1):24-34.e10. doi: 10.1016/j.cels.2019.06.006.
18. Zhang Z, et al. A survey and evaluation of Web-based tools/databases for variant analysis of TCGA data. Brief Bioinform. 2019;20(4):1524-1541. doi: 10.1093/bib/bby023.
19. Pepe G, et al. PANDA: PAN Cancer Data Analysis Web Tool. J Mol Biol. 2025;437(15):169158. doi: 10.1016/j.jmb.2025.169158.
20. Wolde T, et al. Current Bioinformatics Tools in Precision Oncology. MedComm (2020). 2025;6(7):e70243. doi: 10.1002/mco2.70243.
21. National Cancer Institute Genomic Data Commons. Data Transfer Tool Command Line Documentation. Available at: https://docs.gdc.cancer.gov/Data_Transfer_Tool/Users_Guide/Data_Download_and_Upload/. Accessed 2026 Jun 8.
22. Microsoft. Performance Counters - Win32 apps. Microsoft Learn. Available at: https://learn.microsoft.com/en-us/windows/win32/perfctrs/performance-counters-portal. Accessed 2026 Jun 8.
23. Python Software Foundation. concurrent.futures - Launching parallel tasks. Python 3 documentation. Available at: https://docs.python.org/3/library/concurrent.futures.html. Accessed 2026 Jun 8.
24. Salimy S, Lanjanian H, Abbasi K, Salimi M, Najafi A, Tapak L, et al. A deep learning-based framework for predicting survival-associated groups in colon cancer by integrating multi-omics and clinical data. Heliyon. 2023;9(7):e17653. doi: 10.1016/j.heliyon.2023.e17653.
25. Habibzadeh G, Mokhtari K, Heshmati M, Salimy S, Mei Z, Entezari M, et al. Identification of lncRNA associated with the SERPINE1 gene in colorectal cancer through TGF-beta pathway. Comput Biol Med. 2025;190:110037. doi: 10.1016/j.compbiomed.2025.110037.
26. Tapak L, Hamidi O, Amini P, Afshar S, Salimy S, Dinu I. Identification of Prognostic Biomarkers for Breast Cancer Metastasis Using Penalized Additive Hazards Regression Model. Cancer Inform. 2023;22:11769351231157942. doi: 10.1177/11769351231157942.
27. Sabit H, Yadav AK, Salimy S, Sakr A, Abdel-Ghany S, Soliman Wadan A, et al. Integrating multi-omics and artificial intelligence for personalized breast cancer management: A guide to clinicians. Cancer Lett. 2026;649:218468. doi: 10.1016/j.canlet.2026.218468.
28. Haji Molla Hoseyni B, Lanjanian H, Zohrab Beigi Y, Salimi M, Zare-Mirakabad F, Masoudi-Nejad A. Molecular landscape of endometrioid Cancer: Integrating multiomics and deep learning for personalized survival prediction. Comput Biol Med. 2025;192(Pt A):110284. doi: 10.1016/j.compbiomed.2025.110284.
29. Beigi YZ, Lanjanian H, Fayazi R, Salimi M, Haji Molla Hoseyni B, Noroozizadeh MH, et al. Heterogeneity and molecular landscape of melanoma: implications for targeted therapy. Mol Biomed. 2024;5:17. doi: 10.1186/s43556-024-00182-2.
30. Python Software Foundation. tkinter - Python interface to Tcl/Tk. Python 3 documentation. Available at: https://docs.python.org/3/library/tkinter.html. Accessed 2026 Jun 8.
31. Nielsen J, Molich R. Heuristic evaluation of user interfaces. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems; 1990. p. 249-256. doi: 10.1145/97243.97281.
32. Abedi Z, et al. mRNA-miRNA bipartite networks reconstruction in different tissues of bladder cancer based on gene co-expression network analysis. Sci Rep. 2022;12:5885. doi: 10.1038/s41598-022-09920-4.
33. Abedi Z, et al. Metagenomic insights into microbial community alterations and co-occurrence networks in infective endocarditis. Available at:

https://pmc.ncbi.nlm.nih.gov/articles/PMC1267
0860/. Accessed 2026 Jun 20.