NAR Genomics and Bioinformatics, 2026, **8**, lqag029
https://doi.org/10.1093/nargab/lqag029
Pipelines and Workflows for Biological Data Analysis

OXFORD

Eduomics: a Nextflow pipeline to simulate -omics data for education

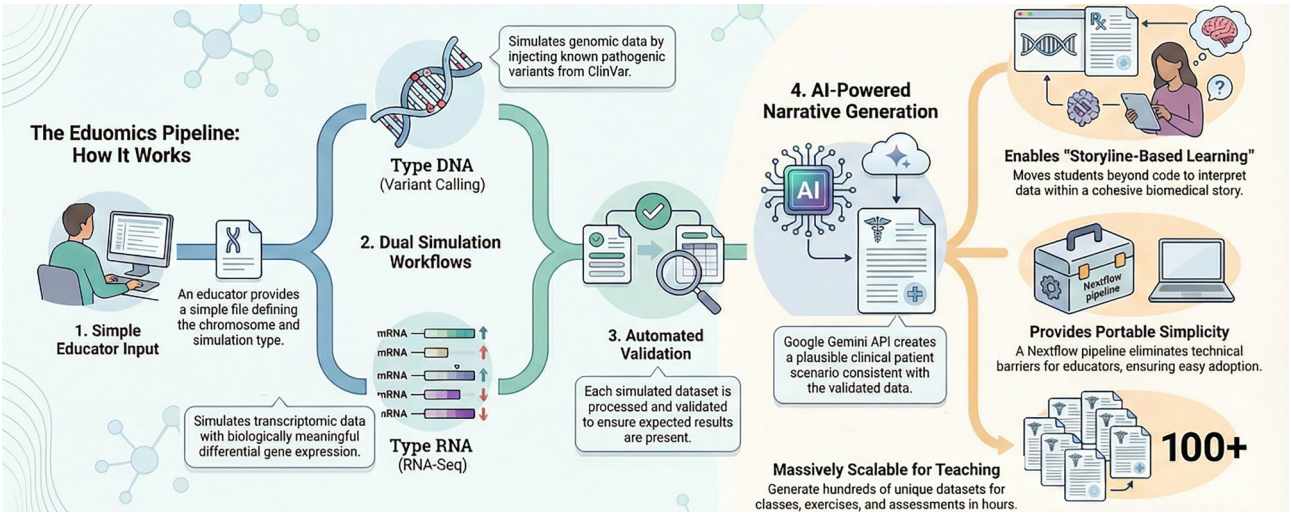
Lorenzo Sola, Davide Bagordo, Simone Carpanzano, Mariangela Santorsola, Francesco Lescai *

Department of Biology and Biotechnology "L. Spallazani", University of Pavia, Pavia, IT-27100, Italy
*To whom correspondence should be addressed. Email: francesco.lescai@unipv.it

Abstract

Bridging the gap between learning algorithms and biological interpretation remains the central challenge of bioinformatics education. To move students beyond just learning code and acquire higher-order knowledge, educators face a complexity overload when attempting to design realistic learning experiences. Simulating realistic data forces educators to master different tools and dependencies. Existing simulators are built for benchmarking and lack the narrative context essential for teaching interpretation; this limitation prevents a shift from problem-solving to what we believe should be an immersive 'storyline-based learning'. Eduomics is a Nextflow pipeline designed to simulate data, create stories, and improve teaching. It provides portable simplicity, by automating the generation of validated genomic and transcriptomic datasets while abstracting away all technical requirements. Uniquely, it leverages the Gemini API to construct clinical narratives consistent with the simulations. Rather than analysing isolated datasets, students engage with a cohesive biomedical story, replicating the cognitive demands of real-world diagnostics. Eduomics facilitates massive scaling: educators can generate hundreds of unique, validated datasets for assessments or tutoring in a matter of hours. By removing barriers to adoption and embedding raw data within a rich clinical context, eduomics offers an accessible, scalable solution to truly innovate bioinformatics education.

Graphical abstract



Introduction

The rapid evolution of next-generation sequencing (NGS) technologies has profoundly reshaped both biomedical research and clinical practice. The continued decrease in sequencing costs, together with the proliferation of multi-omics data, is transforming the life sciences into a data-intensive discipline. As a result, students enrolling in biomedical studies are increasingly expected to manage complex computational workflows: bioinformatics has thus become a cornerstone of modern life sciences education. Consequently, there is a growing consensus that bioinformatics training should be introduced early in higher-education curricula [1–3] to equip future scientists with the computational and analytical competencies required in contemporary laboratories.

In this context, simulated datasets offer a controlled and flexible environment where learners can engage with realistic workflows without raising ethical or privacy concerns. However, using existing tools for data simulation in an

Received: December 9, 2025. **Revised:** February 9, 2026. **Accepted:** February 16, 2026

educational setting remains challenging. Most of these tools are designed for benchmarking software or algorithm development, rather than for educational purposes. An educator faces a steep learning curve, which is a barrier to the adoption of simulations as part of common teaching and learning designs. Their use typically requires choosing the appropriate software, installing their dependencies, generating compatible reference datasets, creating input files in specific formats, and manually integrating intermediate outputs. Even when these obstacles are faced, significant extra work is required to connect the data generated by existing simulators with the underlying molecular mechanisms and the clinical phenomena they participate in. However, embedding the simulations in the appropriate context is necessary to interpret the analysis results. This is an essential skill that educators aim to develop in students. A few examples of this gap can be observed in genomics, where different solutions have been developed to simulate genetic and transcriptomic data. For example, a broad ecosystem of variant-injection and genome-perturbation tools enables users to embed predefined variants into simulated reads or alignment files, thereby providing high-fidelity benchmarks for variant calling and alignment evaluation [4–8]. Similarly, a variety of tools have been developed to simulate RNA sequencing (RNA-seq) reads for benchmarking quantification and differential-expression workflows. This software can model experimental biases such as GC content, transcript length, and positional effects, enabling the creation of datasets with known differentially expressed genes (DEGs) [9–11]. While these tools produce realistic data, their workflow is fragmented and context-dependent: each focuses on a single step of the process, leaving users responsible for chaining outputs, normalizing file formats, and managing parameters. An additional challenge educators face, especially when dealing with data simulations in official curricula, is the need to generate hundreds of datasets with the same characteristics: this is because different datasets are needed not only for classes, but also for exercises, tutoring, and, more importantly, assessments.

Automated, accessible, scalable, and portable solutions are needed to overcome this key challenge. In recent years, a few end-to-end NGS simulation pipelines have emerged, which generate synthetic biological datasets designed to mimic sequencing experiments. These frameworks reproduce empirical features such as sequencing errors, mutation spectra, and gene expression profiles, producing realistic resources for benchmarking, workflow optimization, and software validation [12–14]. Yet, even these comprehensive pipelines remain primarily research- and benchmark-oriented: they offer technical realism without addressing the need, compelling in education, to contextualize the molecular data simulated within biological systems, or even clinical scenarios.

To bridge these key gaps, we developed eduomics: the pipeline is designed to generate hundreds of datasets, enabling the acquisition of both technical proficiency as well as conceptual understanding, and to provide teachers with a single end-to-end tool which transforms complex simulations into an accessible educational opportunity. Eduomics is a Nextflow-based pipeline specifically designed for educational use that automates all key steps of NGS simulation, from reference preparation to results interpretation. Unlike traditional simulators, this pipeline generates biologically coherent variant calling and RNA-seq datasets enriched with clinically validated variants and realistic transcriptomic profiles. Each simulation is defined by a known ground truth, ensuring full control over the injected variants or expression changes and allowing students to trace analytical outcomes back to their biological causes. By integrating Google Gemini's generative capabilities, the pipeline links molecular alterations to AI-derived clinical narratives, creating immersive learning experiences. This educational design goes beyond the traditional problem-based learning by introducing a more complete, scenario-driven framework. Through the combination of validated data, clinical context, and narrative design, this approach establishes a pedagogical framework grounded in storyline-based learning, enabling students to explore the biological significance of computational results in realistic biomedical contexts. Students are more engaged, understand better the significance of their work, and, therefore, are likely more motivated to follow along in the journey.

Materials and methods

Pipeline implementation

Eduomics was implemented using the Nextflow DSL2 language [22], following the modular structure and best-practice guidelines of the nf-core community [23]. The workflow is composed of 16 modules from the shared nf-core repository, and 14 modules specifically developed to enable the pipeline's peculiar functionalities. These latter modules wrap custom scripts implemented in R, Bash, and Python. Software dependencies are resolved using Conda, Docker, or Singularity, ensuring portability and reproducibility across computational environments. Continuous integration tests are executed through GitHub Actions on small datasets, while full-scale benchmarking was carried out on Google Cloud. The eduomics pipeline is openly available at https://github.com/lescai-teaching/eduomics.

Input definition

The pipeline requires a sample sheet in CSV format, used to define the simulation parameters. This file specifies the type of workflow to be executed (variant calling or RNA-seq) and includes optional and mandatory fields. Among the optional parameters, two are particularly relevant: (i) capture, a path to an exome capture BED file, required for variant-calling simulations; and (ii) the similarity threshold, a Jaccard index used to construct similarity networks from Gene Ontology (GO) annotations (see later), applied to RNA-seq simulations. Additional parameters allow users to select the chromosome, sequencing coverage, number of replicates, and grouping of simulated datasets.

Variant calling simulation workflow

Reference sequences from the GRCh38 genome assembly [24] were subset to the user-specified chromosome and indexed using nf-core modules samtools_faidx (samtools v1.21) [25], bwa_index (bwa v0.7.18) [26], and gatk4_createsequencedictionary [27]. Target regions for the simulation were processed with the local module subset-capture, implemented in Bash and relying on bedtools [28] and htslib [29]. This module compressed and indexed the capture BED file, extracted the chromosome-specific intervals, and generated padded regions (±50 and ±500 bp). Reference Variant Call Format (VCF) files from gnomAD,

Mills indels, 1000G, and dbSNP [24] were subset to the same regions using gatk4_selectvariants, providing interval-specific files. ClinVar variants have been downloaded from release GRCh38 in VCF format and were processed with the local module subvar, implemented in Bash with htslib, which standardized chromosome nomenclature, extracted variants overlapping the target regions, and stratified the resulting set into distinct VCF files containing benign or pathogenic records. Together, these steps generated the DNA reference bundle, including a chromosome-specific genome FASTA with the relative index, the bwa index, capture targets, interval-specific VCFs, and curated ClinVar sets, which were used as the basis for read simulation and variant calling. Read simulation was carried out by first generating a profile for a sequencing experiment on the target chromosome with wgsim [30]: this generated paired-end FASTQ files from the chromosome-specific reference FASTA. The simulated reads were aligned to the same reference using bwa-mem, and the resulting BAM files were indexed with samtools. Downstream pre-processing followed the GATK Best Practices guidelines: duplicate reads were identified and marked with gatk4_markduplicates; base quality score recalibration (BQSR) was performed with gatk4_baserecalibrator, using known sites from dbSNP and Mills indels; the recalibrated BAMs were produced with gatk4_applybqsr and re-indexed; variants were called with gatk4_haplotypecaller. Finally, the resulting VCF was passed, together with the recalibrated BAM files and the chromosome-specific FASTA, to the local module simuscop_seqtoprofile. The module, implemented with SimuSCoP v1.1.2 [8], produced a profile file representing sequencing- and variant-related characteristics. The benign and pathogenic variant sets obtained from ClinVar were then converted into the variation format required by SimuSCoP using the local module pyconvertosim. This Python-based step (v3.9) generated three types of files: a baseline variation file containing only benign variants, a pathogenic variation file, and multiple combined variation sets, each including the benign background plus one pathogenic variant. Each combined variation file was then processed with the module simuscop_simureads, which integrates sequencing profiles, reference genome, capture intervals (±500 bp), and the prepared variant sets to simulate case and control samples from the selected chromosome. For each pathogenic variant, paired-end FASTQ files were generated, representing control samples (containing only benign variants) and case samples (containing both benign and the selected pathogenic variant).

Simulated case and control reads were then processed jointly following the GATK Best Practices, from alignment to variant calling, producing single-sample genomic VCFs (gVCFs). To enable joint genotyping across samples, the gVCFs were processed with gatk4_genomicsdbimport and subsequently genotyped with gatk4_genotypegvcfs, generating a combined, multi-sample VCF.

The combined VCF was then compared with the simulated case/control reads using the local module dnavalidation, implemented in Bash. This step verified the presence of the injected pathogenic variant within the simulated dataset by matching its genomic position in the VCF to the expected coordinates. The validated VCF, together with the corresponding simulated reads, was then stored in variant-specific result directories.

RNA-seq simulation workflow

The GRCh38 genome assembly [24] was first subset and indexed to match the target chromosome with the nf-core module samtools_faidx. The genome GFF3 annotation from GENCODE v48 was filtered to retain only protein-coding transcripts from the user-selected chromosome. The annotation file was processed with the local module subsetgff, based on an R script relying on rtracklayer v1.66.0 [31] for GFF3 parsing, AnnotationDbi v1.68.0 [32], and org.Hs.eg.db v3.20.0 [33] to retrieve GO annotations. To generate simulations in which DEGs produced a meaningful enrichment analysis, we selected a subset of transcripts that shared membership across enough ontologies and use these sets to generate the data. To this purpose, a transcript-GO incidence matrix was generated, a pairwise transcript similarity was calculated as a Jaccard index (default 0.3) [34], and a transcript similarity network was constructed with igraph v1.5.1 [35]. Gene communities were identified using the Louvain community detection algorithm, and to ensure biological plausibility, only clusters containing more than three but fewer than 10% of the total genes in the chromosome were retained. The biological relevance of these clusters was subsequently assessed through GO enrichment analysis performed with clusterProfiler v4.14.0 [36], producing a set of gene lists. The reference transcriptome (GENCODE v48 FASTA) was filtered with the local module subsetfastatx, which implements an R script for transcript-annotation matching and uses Biostring v2.74.0 [32] for sequence handling, resulting in a chromosome-specific transcript FASTA.

Synthetic count matrices were then produced with the local module countmatrices, based on in-house R functions. Baseline counts were initialized proportionally to transcript length and user-defined coverage, then fold changes were applied to the GO-selected transcripts to encode coherent up- and down-regulation across groups and replicates. Read-level simulation was carried out with the local module polyester_simulate, which uses the polyester package v1.38.0 [10] to generate synthetic reads from the filtered transcript FASTA. In the current implementation, the module operates in count-matrix mode, receiving the simulated count matrices as input to produce the reads. Although the workflow supports an alternative mode based on simulated fold changes, this functionality is not yet implemented. Simulated reads were produced in gzipped FASTA format and provided as input for downstream analyses. Quantification of simulated reads was performed with the nf-core module salmon_quant using salmon v1.10.3 [37] in mapping-based mode. Gene-level counts were then obtained in R with tximport v1.34.0 [38] by mapping transcripts to genes using the tx2gene file derived from the filtered annotation. Differential expression analysis was carried out through the deanalysis module, implemented using DESeq2 v1.46.0 [39]. The sample design was constructed from the specified numbers of replicates and groups, genes with low counts were pre-filtered, and the condition was modelled using the design formula ~ condition. Standard diagnostic (including MA, dispersion, and counts plot) and exploratory summary plots (principal component analysis - PCA - and heatmap) were generated. The results were exported as a ranked table listing adjusted *P*-values (padj) and $\log_2$ fold changes. Genes were defined as differentially expressed when padj < 0.05. Functional enrichment was performed using the enrichment module with clusterProfiler, testing the different GO ontologies (Biological

Processes - BP, Molecular Functions - MF, and Cellular Components - CC). Summary visualizations (dot plots and cnetplots) were generated for each ontology. A final validation step was implemented in the local module rnaseqvalidation, which relies on an R script using DOSE v4.0.0 [40]. A simulation was classified as successful when at least one ontology (BP, MF, or CC) contained three or more significantly enriched terms. Upon validation, the RNA framework produced a complete set of outputs suitable for downstream training and interpretation. These include: (i) a reference bundle (FASTA transcriptome, index files, annotation GFF3), (ii) simulated reads in gzipped FASTA format, (iii) differential expression results, (iv) functional enrichment results, and (v) a collection of diagnostic and exploratory plots.

AI-based scenario generation

A final module, aiscenarios, is shared by both the variant calling and RNA-seq workflows. This module generates patient-inspired clinical narratives from the simulated data using the Google Gemini API. Depending on the input, the module ingests either pathogenic variants with the corresponding gene from the variant calling workflow (provided with the option '–variant' and the format 'chr22-1234-A-T', and with option '–gene' and format 'TANGO2' to the script) or lists of DEGs from the RNA-seq workflow (provided with the option '–genes' and the format 'BRCA1, TP53, EGFR' to the script). The corresponding Python script, implemented with the google-genai library (v1.9.0) [41], instructed Gemini to generate a detailed clinical scenario describing a hypothetical patient, including relevant clinical features, family history, and diagnostic clues consistent with the underlying molecular and genetic dysfunctions. The output of this module consisted of a text file containing realistic clinical cases, which serve as the final layer of the storyline-based educational framework of eduomics. More details on the prompt used for the two dataset types and an example of the exact formulation used to generate each scenario is given in the Supplementary Materials.

Results

Eduomics is a pipeline designed to simulate genomics data, create stories to accompany the datasets, and to improve teaching. It provides portable simplicity by automating the generation of validated genomic and transcriptomic datasets while abstracting away all technical requirements. The pipeline has been developed using the Nextflow DSL2 language, combining modular processes in line with nf-core community standards. The pipeline integrates 16 modules from the shared nf-core repository and 14 modules specifically developed to enable the pipeline functionalities (called "local", according to the nf-core community nomenclature).

The release 1.1.0 (https://zenodo.org/records/17552833, accessed on 7 November 2025) accepts a sample sheet in CSV format as an input: this file defines the parameters for each simulation, which produces validated variant calling and RNA-seq datasets together with AI-generated clinical scenarios. An overview of the main workflow is shown in Fig. 1, and a detailed pipeline diagram is provided in Supplementary Information (Supplementary Fig. S1).

The variant calling workflow (Fig. 1A) generates synthetic resequencing data by injecting known, likely pathogenic or pathogenic variants from the ClinVar release into a user-defined genomic interval. This workflow generates simulated raw reads from a case/control sample design and processes them according to GATK Best Practices to validate the results. This ensures that the DNA datasets resemble realistic resequencing experiments while embedding variants of clinical relevance.

The RNA-seq workflow (Fig. 1B) simulates transcriptomic data with biologically meaningful differential expression. This workflow prepares the reference files, applies fold changes to selected gene sets generated from common ontologies in GO, and generates synthetic read counts, followed by quantification and enrichment analysis, to validate the simulated data. This design allows the simulation of RNA-seq datasets that reproduce expression patterns consistent with biologically relevant pathways.

By processing both types of simulations through a complete analysis workflow, only those simulated datasets which produce the expected results are saved by the pipeline. Validated results serve as input to a common module that employs the Google Gemini API to generate patient-inspired clinical scenarios. These scenarios represent a key outcome of eduomics, with the goal of bridging the operational understanding of computational tools with the clinical interpretation as a key element of the teaching and learning approach.

To evaluate performance, we benchmarked eduomics on human chromosome 22. The pipeline was tested both on an on-premise HPC system (SLURM scheduler) and on Google Cloud, to demonstrate full portability across computational environments. As a case study, we present here the run on Google Cloud Platform, which allows a more fine-grained report of the workflow performance and resource utilization (Fig. 2), as well as the actual costs. The simulation completed in ∼7 h (Fig. 2A), involving a total of 11 794 submitted jobs (Fig. 2B), and an estimated execution cost of €199.11.

The pipeline comprises a series of well-known memory-intensive tasks, such as GATK HaplotypeCaller for variant calling, GATK MarkDuplicates for identifying and marking duplicate reads, and polyester for RNA-seq read simulation. As shown in Fig. 2C, these components exhibit distinct memory usage profiles reflecting their computational nature. HaplotypeCaller displayed the greatest variability, with memory consumption ranging from ∼2 to 6 GB, consistent with the dynamic allocation required for the local de novo assembly performed during variant calling. MarkDuplicates maintained a stable peak around 5 GB, corresponding to the handling of large BAM files during duplicate identification. Similarly, polyester required around 4.5 GB of memory. Overall, these results confirm that eduomics efficiently manages a large number of parallel jobs across multiple compute nodes: the parallelism of the execution is visible in the Gantt Chart (Fig. 2A), and the number of jobs running at the same time can be appreciated in Fig. 2B. This ensures stable performance and smooth execution despite the complexity of the pipeline (Supplementary Information).

The variant calling workflow achieved a validation rate of 38.2%, validating 241 out of 631 injected pathogenic variants. The RNA-seq workflow reached a validation rate of 78.5%, with 11 out of 14 differential expression patterns successfully recovered after quantification and differential expression analysis (Supplementary Information and Supplementary Fig. S2). Figure 3 provides two illustrative examples of simulations generated from the benchmark on chromosome 22.

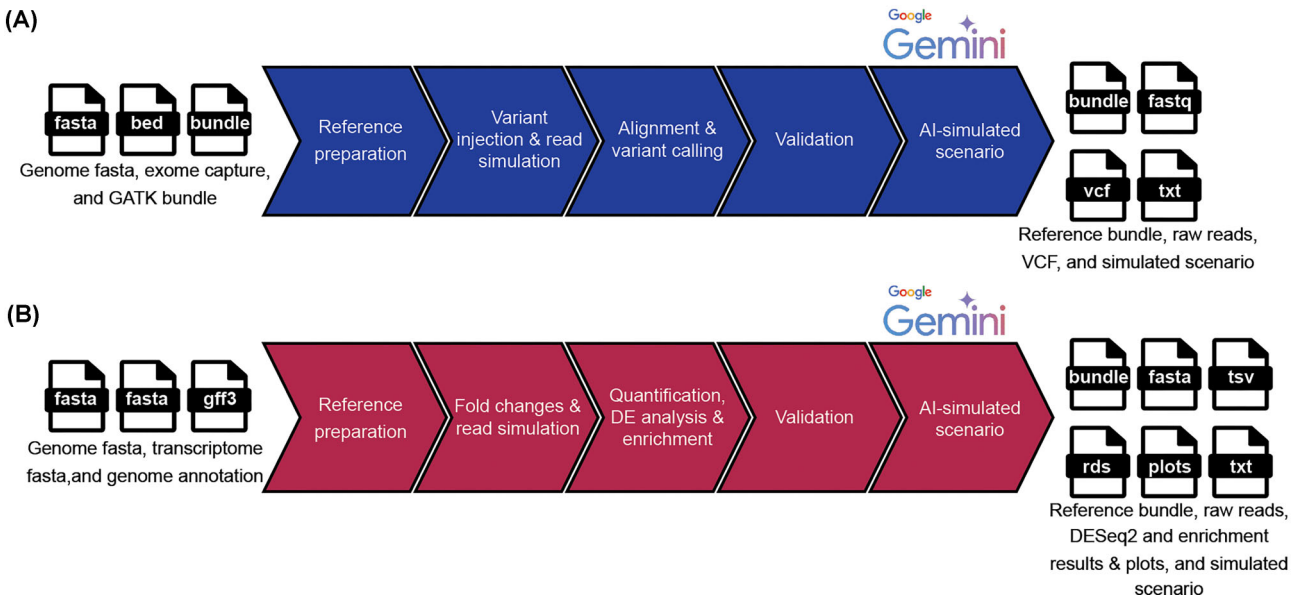


Figure 1. Overview of the eduomics pipeline. The pipeline comprises two main workflows: (**A**) the variant calling workflow, which simulates resequencing data with selected likely pathogenic and pathogenic variants, and (**B**) the RNA-seq workflow, which generates transcriptomic datasets with biologically meaningful differential expression. The outputs produced from each workflow converge into a final common module that employs Google Gemini to generate patient-inspired clinical scenarios. The arrows illustrate the logical progression of each workflow, from initial inputs to final outputs.

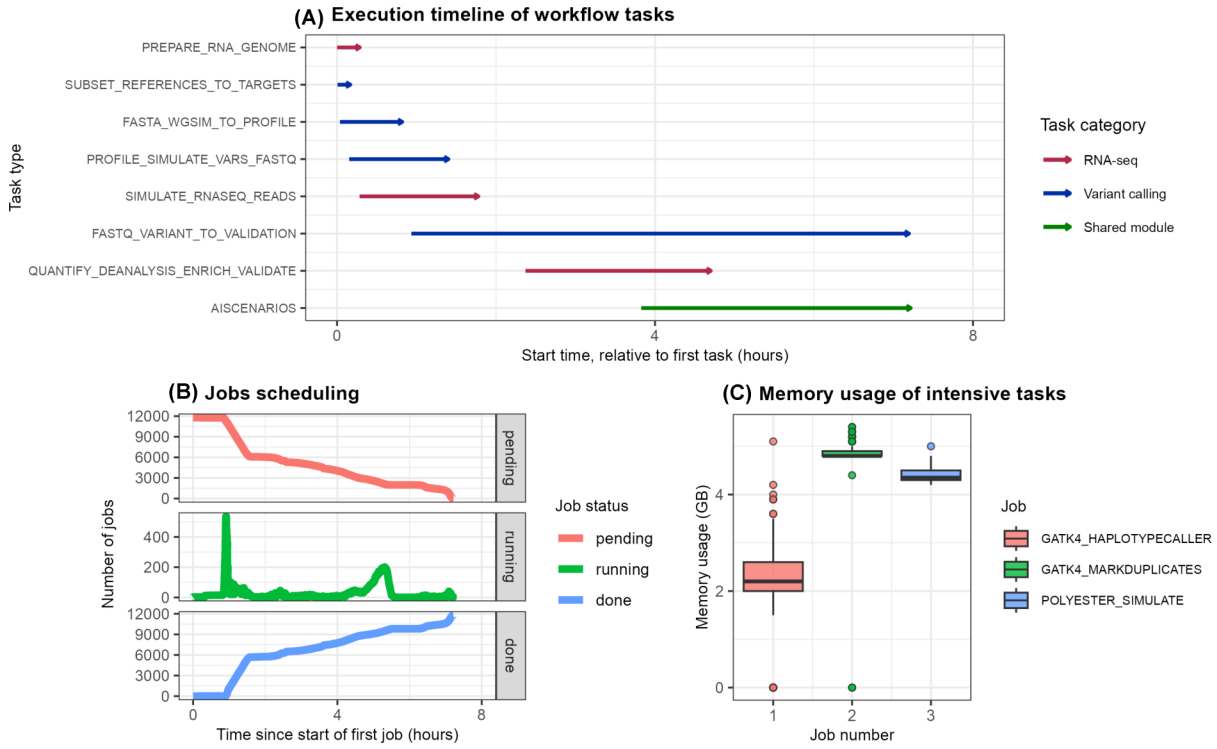


Figure 2. Progress and performance of the benchmark analysis on chromosome 22. The figure represents different performance indicators for a run of eduomics tasked to produce both variant calling and RNA-seq datasets. (**A**) Gantt plot showing the execution timeline of eduomics tasks. Each bar represents the start and completion time of individual tasks. (**B**) The plot represents the progress of the analysis on Google Cloud, illustrating the dynamic status of tasks (pending, running, completed) over time. (**C**) Boxplot summarizing memory utilization for the top three most memory-intensive steps in the workflow.

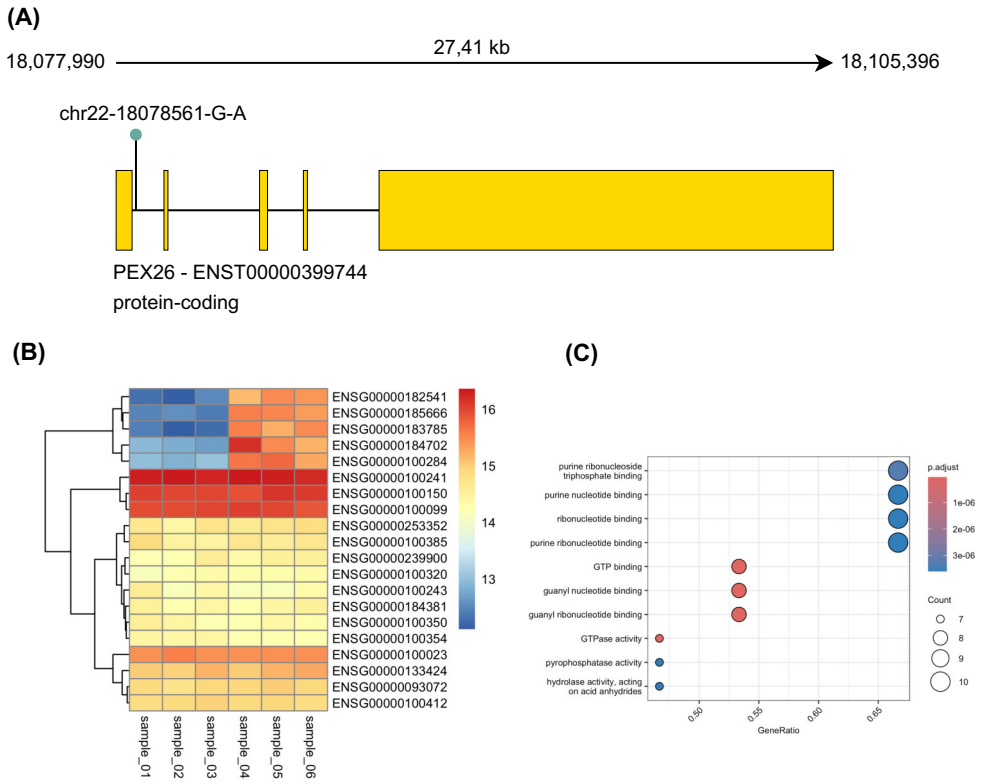


Figure 3. Exemplificative results from the benchmark analysis on chromosome 22. (**A**) Example from the DNA variant calling workflow. The diagram is a 'lollipop plot', commonly used in medical genetics to highlight where a mutation might hit a relevant gene, and shows the *PEX26* gene (ENST00000399744) located on chromosome 22. In this simulation, a stop-gained mutation was introduced at position 18078561 (G > A). (**B**) Example from the RNA-seq workflow. Heatmap of the 20 most highly expressed genes across simulated samples, with colours ranging from blue (lower expression) to red (higher expression). (**C**) Standard dot-plot representing the results of the over-representation analysis carried out on DEGs against GO biological processes, where the gene ratio represents the proportion of DEGs contributing to the enrichment versus the genes annotated to the ontology.

Figure 3A illustrates an example from the variant calling workflow. The case corresponds to the injected stop-gained mutation chr22-18078561-G-A in the *PEX26* gene (ENST00000399744), resulting in a premature termination codon (p.Trp62Ter). This variant is annotated as pathogenic/likely pathogenic in ClinVar. In the simulated dataset, aligned reads from the case sample display an adenine (A) at position 18078561, in contrast to the guanine (G) present in both the control sample and the human reference genome.

The affected codon lies within the N-terminal region of the peroxin-26 protein, a peroxisomal membrane anchor essential for the assembly and biogenesis of peroxisomes. Truncating mutations in *PEX26* are known to disrupt peroxisome biogenesis, leading to defective import of matrix proteins and the accumulation of very long-chain fatty acids (VLCFAs) [15]. The simulated clinical scenario (Supplementary Information, Scenario 1 and 2) generated by eduomics reproduces a coherent genotype–phenotype correlation consistent with a peroxisome biogenesis disorder. The patient presents signs of developmental delay, hypotonia, and early-onset pigmentosa. Consistent with the known literature on the disease [16, 17], the generated clinical scenario describes biochemical investigations showing elevated VLCFA levels, which, together with the neurological symptoms, visual impairment, and liver involvement, support the diagnosis of a peroxisomal biogenesis disorder. The integration of variant calling with the generation of a phenotypic and clinical history thus exemplifies the educational potential of the workflow in demonstrating the molecular link between a pathogenic variant and a real-world clinical scenario.

An example of the validated simulations generated by the RNA-seq workflow is presented in Fig 3B. The heatmap displays the 20 most highly expressed genes, highlighting a clear separation between control samples (sample 01–03) and case samples (sample 04–06). Notably, five genes (ENSG00000182541, ENSG00000185666, ENSG00000183785, ENSG00000184702, ENSG00000100284) show a particular differential expression profile between conditions. These were subsequently identified as significantly differentially expressed, together with the complete set of significantly DEGs reported in Supplementary Table S1. The functional enrichment analysis revealed a strong over-representation of molecular functions related to nucleotide binding and GTPase activity (padj $< 10^{-6}$), reflecting the involvement of genes in processes such as energy metabolism and intracellular signalling, which are frequently dysregulated in neurodevelopmental and skeletal disorders [18, 19]. These findings support the biological coherence of the simulated dataset, which served as the molecular foundation for eduomics to generate an AI-derived clinical scenario. An example of such scenario is reported in Supplementary Information, illustrating a patient affected by progressive neurological decline, developmental

delay, and skeletal abnormalities. The diagnostic investigation described in Gemini's generated scenario revealed diffuse cerebral atrophy, ataxia, and hypotonia, pointing to a complex neurological and skeletal disorder. In this AI clinical picture, whole-genome sequencing was also included among the diagnostic investigations to further elucidate the underlying molecular cause, mirroring real-world clinical practice. Similarly to the variant calling workflow, the integration of transcriptomic profiling with an AI-generated medical history exemplifies the educational potential of the RNA-seq workflow in linking altered gene expression patterns to clinically interpretable scenario. A detailed example of classroom implementation is provided as part of the Supplementary Material. Other AI solutions were explored, but Gemini offered the higher API rates in the free tier, and therefore our choice was largely driven by accessibility, i.e. offering the opportunity to educators to use all eduomics feature regardless of the availability of a licence to a particular AI provider.

Discussion

Problem-based learning supported by data simulations provides students with the opportunity to practise data analysis in realistic and controlled environments. At the same time, such resources allow instructors to promote active learning through hands-on experience. Widespread introduction of simulated datasets in bioinformatics education is, however, hampered by several challenges. From a technical perspective, there are significant barriers to adoptions: steep learning curves, specific requirements each tool has, difficulties in deployment, lack of automation, combined with the need to generate hundreds of datasets in contexts where simulated data serve also for tutoring and assessment [4–14]. From a pedagogical perspective, learning bioinformatics goes beyond acquiring methodological knowledge; however, we argue that problem-based learning alone is not sufficient to achieve long-term professional proficiency. According to the structured of the observed learning outcome taxonomy [20], higher-level intended learning outcomes involve the development of cognitive processes described by verbs such as 'explain', 'theorise', 'hypothesise' [20, 21]: achieving these outcomes requires students not only to move beyond the acquisition of procedural skills, but also beyond the application of specific tools to different questions, and demonstrate instead a broader conceptual understanding. To reach these goals, data simulations must be integrated with a wider educational context that provides a clear purpose, an end goal, and an authentic motivation for the students to learn technical skills and apply them to real-life scenarios. At the same time, this context should train students to interpret their results in light of the bigger picture and to reach original conclusions. This pedagogical strategy creates a narrative framework that helps students understand not only how to perform analyses, but also the intrinsic behaviour of biological data they explore within a realistic context. We called this storyline-based learning.

Eduomics has the ambition to introduce a disruptive innovation in bioinformatics education by addressing these challenges and enabling such a holistic approach to teaching and learning. By developing this Nextflow-based pipeline and automate all steps of NGS simulations, we demonstrate that realistic genomic and transcriptomic simulations can be made accessible to bioinformatics educators without prior knowledge of simulation software, and can be deployed in a scalable way, producing pedagogically meaningful datasets. With this solution, instructors are only required to define a chromosome and select the type of simulation; this approach eliminates technical entry barriers, and allows the adoption of a scenario-based teaching design with appropriate datasets. By generating hundreds of data, combined with unique clinical scenarios, educators can assign unique datasets to each student or group, creating personalized learning experiences while maintaining reproducibility and consistency. This design supports a wide scale of teaching opportunities, from small-group workshops to large online courses. In addition, because each dataset is biologically validated against a known ground truth, instructors retain the full control over the simulated events and can ensure that analytical outcomes are always traceable back to their molecular causes. Thanks to its Nextflow implementation, eduomics is fully portable across infrastructures and can be deployed both on local high-performance clusters and in cloud environments at minimal cost. This ensures that high-quality simulations are affordable and accessible to educational institutions regardless of their computational resources.

Despite all these advantages, the current version of eduomics still presents a few limitations, which will be addressed in future releases. The number of simulations produced by the variant-calling workflow is significantly higher compared to those generated by the RNA-seq workflow. This difference is partly expected, as the simulations were limited to a single chromosome. This constraint is necessary to keep the resulting data files sufficiently small to be shared, and the run times sufficiently short to fit into the duration of a lecture. While each single variant can produce a simulation, genes must be selected in sets to simulate meaningful differential expression patterns. Therefore, the number of variants per chromosome is inherently larger than the number of genes sets which can be generated, and produce valid functional enrichments. The complexity of generating transcriptomic profiles is constrained by the gene ontologies of the selected chromosome. A key parameter influencing the outcome of the RNA-seq simulations is the similarity threshold (default = 0.3). By fine-tuning this parameter across different runs of the pipeline, users can modulate the number of the generated RNA-seq datasets. This limitation might be partially addressed by including other annotations (KEGG, Reactome) in addition to GO. The validation rates are different for the variant-calling and the RNA-seq data due to the design of the simulations: while pathogenic variants are injected as reported in ClinVar, and simulated data are validated to assess whether the injected variant has been called at the end of the workflow, DEGs are carefully chosen using ontologies before reads are simulated, and therefore there are fewer variables impacting their validation with enrichment criteria. The validation rate of the variant calling is acceptable, considering that read error profile, coverage variability, and GATK defaults are the main causes, and that a larger number of simulations is produced.

An additional limitation might be related to the quality of the AI-generated scenario: our tests have consistently produced high-quality output, which is certainly beyond the scope the scenario is meant to be used in (context and clues). The end user should, however, always check the text for potential hallucinations, and play a role in choosing more suitable scenarios for their teaching context. We have also found that variant-calling scenarios provide more consistent and easier-to-interpret clues compared to transcriptomics scenarios: this

is due to the inherently more straightforward relationship between a clinically relevant mutation and a disease, compared to how a differentially expressed pathways might contribute to a pathological phenotype.

In summary, eduomics bridges the gap between mastering computational tools and understanding their biological implications. It transforms complex simulation workflows into guided, narrative-driven learning experiences that allow educators to deliver, and students to appreciate, the richness and complexity of biomedical data. By combining automation with biological and clinical storytelling, eduomics redefines bioinformatics training into an engaging, creative, and intellectually rewarding experience for the next generation of scientists.

Acknowledgements

We acknowledge Harshil Patel, Phil Ewels, Friederike Hanssen, Gavin Mackenzie, Jeremy Guntoro, Kevin Menden, Matthias De Smet, Maxime U. Garcia, Michael J. Cipriano, Patrick Hüther, Priyanka Surana, Ramprasad Neethiraj, Santiago Revale, Suzanne Jin, and the nf-core community for developing community modules used in eduomics.

Author contributions: Lorenzo Sola (Formal analysis [equal], Methodology [equal], Software [equal], Visualization [equal], Writing – original draft [equal], Writing – review & editing [equal]), Davide Bagordo (Methodology [equal], Software [equal], Writing – review & editing [equal]), Simone Carpanzano (Methodology [equal], Software [equal], Visualization [equal]), Mariangela Santorsola (Software [equal]), and Francesco Lescai (Conceptualization [equal], Methodology [equal], Software [equal], Writing – original draft [equal], Writing – review & editing [equal])

Supplementary data

Supplementary data is available at NAR Genomics & Bioinformatics online.

Conflict of interest

None declared.

Funding

Support for this work has been provided within the context of the "Innovative Teaching Grant" of the University of Pavia (grant no. 791/2023, prot. no. 157873, dated 18/09/2023).

Data availability

The code used in this study is publicly available on our GitHub repository https://github.com/lescai-teaching/eduomics. A DOI has been assigned to the repository via Zenodo: 10.5281/zenodo.15835069.

References

1. Sayres MAW, Hauser C, Sierk M *et al.* Bioinformatics core competencies for undergraduate life sciences education. *PLoS One* 2018;13:e0196878. https://doi.org/10.1371/journal.pone.0196878
2. Attwood TK, Blackford S, Brazas MD *et al.* A global perspective on evolving bioinformatics and data science training needs. *Brief Bioinform* 2019;20:398–404. https://doi.org/10.1093/bib/bbx100
3. Marangoni R, Bevilacqua V, Cannataro M *et al.* An overview of bioinformatics courses delivered at the academic level in Italy: reflections and recommendations from BITS. *PLoS Comput Biol* 2023;19:e1010846. https://doi.org/10.1371/journal.pcbi.1010846
4. Huang W, Li L, Myers JR *et al.* ART: a next-generation sequencing read simulator. *Bioinformatics* 2012;28:593–4. https://doi.org/10.1093/bioinformatics/btr708
5. Mu JC, Mohiyuddin M, Li J *et al.* VarSim: a high-fidelity simulation and validation framework for high-throughput genome sequencing with cancer applications. *Bioinformatics* 2015;31:1469–71. https://doi.org/10.1093/bioinformatics/btu828
6. Ewing AD, Houlahan KE, Hu Y *et al.* Combining tumor genome simulation with crowdsourcing to benchmark somatic single-nucleotide-variant detection. *Nat Methods* 2015;12:623–30. https://doi.org/10.1038/nmeth.3407
7. Xia LC, Ai D, Lee H *et al.* SVEngine: an efficient and versatile simulator of genome structural variations with features of cancer clonal evolution. *Gigascience* 2018;7:giy081. https://doi.org/10.1093/gigascience/giy081
8. Yu Z, Du F, Ban R *et al.* SimuSCoP: reliably simulate Illumina sequencing data based on position and context dependent profiles. *BMC Bioinformatics* 2020;21:331. https://doi.org/10.1186/s12859-020-03665-5
9. Griebel T, Zacher B, Ribeca P *et al.* Modelling and simulating generic RNA-seq experiments with the flux simulator. *Nucleic Acids Res* 2012;40:10073–83. https://doi.org/10.1093/nar/gks666
10. Frazee AC, Jaffe AE, Langmead B *et al.* Polyester: simulating RNA-seq datasets with differential transcript expression. *Bioinformatics* 2015;31:2778–84. https://doi.org/10.1093/bioinformatics/btv272
11. Lahens NF, Brooks TG, Sarantopoulou D *et al.* CAMPAREE: a robust and configurable RNA expression simulator. *BMC Genomics* 2021;22:692. https://doi.org/10.1186/s12864-021-07934-2
12. Stephens ZD, Hudson ME, Mainzer LS *et al.* Simulating next-generation sequencing datasets from empirical mutation and sequencing models. *PLoS One* 2016;11:e0167047. https://doi.org/10.1371/journal.pone.0167047
13. Miller TLA, Conceição HB, Mercuri RL *et al.* Sandy: a user-friendly and versatile NGS simulator to facilitate sequencing assay design and optimization [Internet]. bioRxiv, https://doi.org/10.1101/2023.08.25.554791, 28 August 2023, preprint; not peer reviewed.
14. Brooks TG, Lahens NF, Mrčela A *et al.* BEERS2: rNA-seq simulation through high fidelity *in silico* modeling. *Briefings in Bioinformatics* 2024;25:bbae164. https://doi.org/10.1093/bib/bbae164
15. Wanders RJA. Metabolic and molecular basis of peroxisomal disorders: a review. *Am J Med Genet A* 2004;126A:355–75. https://doi.org/10.1002/ajmg.a.20661
16. Matsumoto N, Tamura S, Furuki S *et al.* Mutations in novel peroxin gene PEX26 that cause peroxisome-biogenesis disorders of complementation group 8 provide a genotype–phenotype correlation. *Am J Hum Genet* 2003;73:233–46. https://doi.org/10.1086/377004
17. Ebberink MS, Mooijer PAW, Gootjes J *et al.* Genetic classification and mutational spectrum of more than 600 patients with a Zellweger syndrome spectrum disorder. *Hum Mutat* 2011;32:59–69. https://doi.org/10.1002/humu.21388
18. Stankiewicz TR, Linseman DA. Rho family GTPases: key players in neuronal development, neuronal survival, and neurodegeneration. *Front Cell Neurosci* 2014;8:314. https://doi.org/10.3389/fncel.2014.00314
19. Rodríguez-Fdez S, Bustelo XR. Rho GTPases in skeletal muscle development and homeostasis. *Cells* 2021;10:2984. https://doi.org/10.3390/cells10112984

20. Boulton-Lewis GM. The SOLO taxonomy as a means of shaping and assessing learning in higher education. *High Educ Res Dev* 1995;14:143–54. https://doi.org/10.1080/0729436950140201
21. Biggs J, Tang C, Kennedy G. *Teaching for Quality Learning at University* 5th edn. McGraw-Hill Education (UK), 2022, 410, ISBN 9780335250820.
22. Di Tommaso P, Chatzou M, Floden EW *et al*. Nextflow enables reproducible computational workflows. *Nat Biotechnol* 2017;35:316–9. https://doi.org/10.1038/nbt.3820
23. Ewels PA, Peltzer A, Fillinger S *et al*. The nf-core framework for community-curated bioinformatics pipelines. *Nat Biotechnol* 2020;38:276–8. https://doi.org/10.1038/s41587-020-0439-x
24. AWS-iGenomes: Documentation and description of AWS iGenomes S3 resource [Internet]. [cited 17 November 2025]. Available from https://github.com/ewels/AWS-iGenomes
25. Danecek P, Bonfield JK, Liddle J *et al*. Twelve years of SAMtools and BCFtools. *Gigascience* 2021;10:giab008. https://doi.org/10.1093/gigascience/giab008
26. Li H, Durbin R. Fast and accurate short read alignment with Burrows–Wheeler transform. *Bioinformatics* 2009;25:1754–60. https://doi.org/10.1093/bioinformatics/btp324
27. McKenna A, Hanna M, Banks E *et al*. The Genome Analysis Toolkit: a MapReduce framework for analyzing next-generation DNA sequencing data. *Genome Res* 2010;20:1297–303. https://doi.org/10.1101/gr.107524.110
28. Quinlan AR, Hall IM. BEDTools: a flexible suite of utilities for comparing genomic features. *Bioinformatics* 2010;26:841–2. https://doi.org/10.1093/bioinformatics/btq033
29. Bonfield JK, Marshall J, Danecek P *et al*. HTSlib: c library for reading/writing high-throughput sequencing data. *Gigascience* 2021;10:giab007. https://doi.org/10.1093/gigascience/giab007
30. Li H. lh3/wgsim [Internet]. 2025; [cited 31 October 2025]. Available from https://github.com/lh3/wgsim
31. Lawrence M, Gentleman R, Carey V. rtracklayer: an R package for interfacing with genome browsers. *Bioinformatics* 2009;25:1841–2. https://doi.org/10.1093/bioinformatics/bt⟨?PMU ?⟩p328
32. Huber W, Carey VJ, Gentleman R *et al*. Orchestrating high-throughput genomic analysis with Bioconductor. *Nat Methods* 2015;12:115–21. https://doi.org/10.1038/nmeth.3252
33. Bioconductor. org.Hs.eg.db [Internet]. [cited 31 October 2025]. Available from http://bioconductor.org/packages/org.Hs.eg.db
34. Prokopenko D, Hecker J, Silverman EK *et al*. Utilizing the Jaccard index to reveal population stratification in sequencing data: a simulation study and an application to the 1000 Genomes Project. *Bioinformatics* 2016;32:1366–72. https://doi.org/10.1093/bioinformatics/btv752
35. Csárdi G, Nepusz T. The igraph software package for complex network research [Internet]. *Complex Systems, InterJournal* 2006; [cited 31 October 2025]. p. 1695. Available from https://igraph.org
36. Yu G, Wang LG, Han Y *et al*. clusterProfiler: an R package for comparing biological themes among gene clusters. *OMICS* 2012;16:284–7. https://doi.org/10.1089/omi.2011.0118
37. Patro R, Duggal G, Love MI *et al*. Salmon provides fast and bias-aware quantification of transcript expression. *Nat Methods* 2017;14:417–9. https://doi.org/10.1038/nmeth.4197
38. Soneson C, Love MI, Robinson MD. Differential analyses for RNA-seq: transcript-level estimates improve gene-level inferences. *F1000Res* 2015;4:1521. https://doi.org/10.12688/f1000research.7563.1
39. Love MI, Huber W, Anders S. Moderated estimation of fold change and dispersion for RNA-seq data with DESeq2. *Genome Biol* 2014;15:550. https://doi.org/10.1186/s13059-014-0550-8
40. Yu G, Wang LG, Yan GR *et al*. DOSE: an R/bioconductor package for disease ontology semantic and enrichment analysis. *Bioinformatics* 2015;31:608–9. https://doi.org/10.1093/bioinformatics/btu684
41. Google APIs. *googleapis/Python-genai [Internet]*. [cited 31 October 2025]. Available from https://github.com/googleapis/python-genai

Received: December 9, 2025. **Revised:** February 9, 2026. **Accepted:** February 16, 2026