Reproducibility standards for machine learning in the life sciences

To make machine-learning analyses in the life sciences more computationally reproducible, we propose standards based on data, model and code publication, programming best practices and workflow automation. By meeting these standards, the community of researchers applying machine-learning methods in the life sciences can ensure that their analyses are worthy of trust.

Benjamin J. Heil, Michael M. Hoffman, Florian Markowetz, Su-In Lee,
Casey S. Greene and Stephanie C. Hicks

The field of machine learning has grown tremendously within the past ten years. In the life sciences, machine-learning models are rapidly being adopted because they are well suited to cope with the scale and complexity of biological data. However, there are drawbacks to using such models. For example, machine-learning models can be harder to interpret than simpler models, and this opacity can obscure learned biases. If we are going to use such models in the life sciences, we will need to trust them. Ultimately all science requires trust[1]—no scientist can reproduce the results from every paper they read. The question, then, is how to ensure that machine-learning analyses in the life sciences can be trusted.

One attempt at creating trustworthy analyses with machine-learning models revolves around reporting analysis details such as hyperparameter values, model architectures and data-splitting procedures. Unfortunately, such reporting requirements are insufficient to make analyses trustworthy. Documenting implementation details without making data, models and code publicly available and usable by other scientists does little to help future scientists attempting the same analyses and less to uncover biases. Authors can only report on biases they already know about, and without the data, models and code, other scientists will be unable to discover issues post hoc.

For machine-learning models in the life sciences to become trusted, scientists must prioritize computational reproducibility[2]. That is to say that third parties should be able to obtain the same results as the original authors by using their published data, models and code. By doing so, researchers can ensure the accuracy of reported results and detect biases in the models.

Analyses and models that are reproducible by third parties can be examined in depth and, ultimately, become worthy of trust. To that end, we believe the life sciences community should adopt norms and standards that underlie reproducible machine learning research.

Table 1 | Proposed reproducibility standards

	Bronze	Silver	Gold
Data published and downloadable	x	x	x
Models published and downloadable	x	x	x
Source code published and downloadable	x	x	x
Dependencies set up in a single command		x	x
Key analysis details recorded		x	x
Analysis components set to deterministic		x	x
Entire analysis reproducible with a single command			x

The menu

While many regard the computational reproducibility of a work as a binary property, we prefer to think of it on a sliding scale[2] that reflects the time needed to reproduce. Published works fall somewhere on this scale, which is bookended by 'forever', for a completely irreproducible work, and 'zero', for a work where one can automatically repeat the entire analysis with a single keystroke. As in many cases it is difficult to impose a single standard that divides work into 'reproducible' and 'irreproducible', we instead propose a menu of three standards with varying degrees of rigor for computational reproducibility (Table 1):

1. Bronze standard. The authors make the data, models and code used in the analysis publicly available. The bronze standard is the minimal standard for reproducibility. Without data, models and code, it is not possible to reproduce a work.
2. Silver standard. In addition to meeting the bronze standard: (1) the dependencies of the analysis can be downloaded and installed in a single command; (2) key details for reproducing the work are documented, including the order in which to run the analysis scripts, the operating system used and system resource requirements; and (3) all random components in the analysis are set to be deterministic. The silver standard is a midway point between minimal availability and full automation. Works that meet this standard will take much less time to reproduce than ones only meeting the bronze standard.
3. Gold standard. The work meets the silver standard, and the authors make the analysis reproducible with a single command. The gold standard for reproducibility is full automation. When a work meets this standard, it will take little to no effort for a scientist to reproduce it.

While reporting has become a recent area of focus[3–5], excellent reporting can look akin to a nutritional information panel. It describes information about a work, but it is insufficient for reproducing the work. In the best case scenario, it provides a summary of what the researchers who conducted the analysis know about biases in the data, model limitations and other elements. However, it often does not provide enough information for someone to fully understand how the model came to be. For these reasons, we believe concrete standards for ensuring reproducibility should be preferred over reporting requirements.

Bronze

Data. Data are a fundamental component of analyses. Without data, models cannot be trained and analyses cannot be reproduced. Moreover, biases and artifacts in the data that were missed by the authors cannot be discovered if the data are never made available. For the data in an analysis to be trusted, they must be published.

To that end, all datasets used in a publication should be made publicly available when their corresponding manuscript is first posted as a preprint or published by a peer-reviewed journal. Specifically, the raw form of all data used for the publication must be published. The way the bronze standard should be met depends on the data used. Authors should deposit new data in a specialist repository designed for that kind of data[6], when possible. For example, one may deposit gene expression data in the Gene Expression Omnibus[7] or microscopy images in the BioImage Archive[8]. If no specialist repository for that data type exists, one should instead use a generalist repository like Zenodo (https://zenodo.org) for datasets of up to 50 GB or Dryad (https://datadryad.org/) for datasets larger than 50 GB. When researchers use existing datasets, they must include the information and code required to download and pre-process the data.

Models. Sharing trained models is another critical component for reproducibility. Even if the code for an analysis were perfectly reproducible and required no extra scientist time to run, its corresponding model would still need to be made publicly available. Requiring people who wish to use a method on their own data to re-train a model slows the progress of science, creates an unnecessary barrier to entry and wastes the compute and effort of future researchers. Being unable to examine a model also makes trusting it difficult. Without access to the model, it is hard to say whether the model fails to generalize to other datasets, fails to make fair decisions across demographic groups such as age, sex and nationality, or learns to make predictions based on artifacts in the data.

Because of the importance of sharing trained models, meeting the bronze standard of reproducibility requires that authors deposit trained weights for the models used to generate their results in a public repository. However, authors do not need to publish the weights for additional models from a hyperparameter sweep if one can reproduce the results without them. When a relevant specialist model zoo such as Kipoi[9] or Sfaira[10] exists, authors should deposit the models there. Otherwise, authors can deposit the models in a generalist repository such as Zenodo. Making models available solely on a non-archived website, such as a GitHub project, does not fulfill this requirement.

Source code. From a reproducibility standpoint, a work's source code is as critical as its methods section. Source code contains implementation details that a future author is unlikely to replicate exactly from methods descriptions and reporting tables. These small deviations can lead to different behavior between the original work and the reproduced one. That is, of course, ignoring the huge burden of having to re-implement the entire analysis from scratch. For the computational components of a study, the code is likely to be a better description of the work than the methods section itself. As a result, computational papers without published code should meet similar skepticism to papers without methods sections.

To meet the bronze standard, authors must deposit code in a third-party, archivable repository such as Zenodo. This includes the code used in training, tuning and testing models, creating figures, processing data and generating the final results. One good way of meeting the bronze standard involves creating a GitHub project and archiving it in Zenodo. Doing so gives both the persistence of Zenodo required by scholarly literature and GitHub's resources for further development and use, such as the user support forum provided by GitHub Issues.

Silver

While it is possible to reproduce an analysis with only its data, models and code, this task is by no means easy. Fortunately, there are best practices from the field of software engineering that can make reproducing analyses easier by simplifying package management, recording analysis details and controlling randomness.

One roadblock that appears when attempting to reproduce an analysis stems from differences in behavior between versions of packages used in the analysis. Analyses that once worked with specific dependency versions can stop working altogether with later versions. Guessing which version one must use to reproduce an analysis—or even to get it to run at all—can feel like playing a game of 'package Battleship'. Proper use of dependency management tools such as Packrat (https://rstudio.github.io/packrat/) and Conda (https://conda.io/) can eliminate these difficulties both for the authors and others seeking to build on the work by tracking which versions of packages are used.

Authors may also wish to consider containerization for managing dependencies. Container systems such as Docker[11] allow authors to specify the system state in which to run their code more precisely than just versions of key software packages. Containerization provides better guarantees of reproducing a precise software environment, but this very fact can also facilitate code that will not tolerate even modest environment changes. That brittleness can make it more difficult for future researchers to build on the original analysis. Therefore, we recommend that authors using containers also ensure that their code works on the latest version of at least one operating system distribution. Furthermore, containers do not fully insulate the running environment from the underlying hardware. Authors expecting bit-for-bit reproducibility from their containers may find that graphics processing unit (GPU)-accelerated code fails to yield identical results on other machines due to the presence of different hardware or drivers.

Knowing the steps to run an analysis is a crucial part of reproducing it, yet this knowledge is often not formally recorded. It takes far less time for the original authors to document factors such as the order of analysis components or information about the computers used than for a third-party analyst attempting to reproduce the work to determine that information on their own. Accordingly, the silver standard requires that authors record the order in which one should run their analysis components, the operating system version used to produce the work and the time taken to run the code. Authors must also list the system resources that yielded that time, such as the model and number of central processing units (CPUs) and GPUs and the amount of CPU RAM and GPU RAM required. Authors may record the order in which one should run components: (1) in a README file within the code repository; (2) by adding numbers to the beginning of each script's name to denote their order of execution; or (3) by providing a script to run them in order. Authors must include details on the operating system, wall clock and CPU running time, and system resources used both within the body of the manuscript and in the README.

Box 1 | Aligning reproducibility incentives

Journals. Journals can enforce reproducibility standards as a condition of publication. The bronze standard should be the minimal standard, although some journals may wish to differentiate themselves by setting higher standards. Such journals may require the silver or gold standards for all manuscripts, or for particular classes of articles such as those focused on analysis. If journals act as the enforcing body for reproducibility standards, they can verify that the standards are met by either requiring reviewers to report which standards the work meets or by including a special reproducibility reviewer to evaluate the work.

Badging. A badge system that indicates the trustworthiness of work could incentivize scientists to progress to higher standards of reproducibility. Following completion of analyses, authors could submit their work to a badging organization that would then verify which standards of reproducibility their work met and assign a badge accordingly. Such an organization would probably operate in a similar way to the Bioconductor[18] package review process. Authors could then include the badge with a publication or preprint to tout the effort the authors put in to ensure their code was reproducible. Including these badges in biosketches or CVs would make it simple to demonstrate a researcher's track record of achieving high levels of reproducibility. This would provide a powerful signal to funding agencies and their reviewers that a researcher's strengths in reproducibility would maximize the results of the investment made in a project. Universities could also promote reproducibility by explicitly requiring a track record of reproducible research in faculty hiring, annual review and promotion.

Reproducibility collaborators. Adding 'reproducibility collaborators' to manuscripts would also provide another means to make analyses more reproducible. We envision a reproducibility collaborator as someone outside the primary authors' research groups who certifies that they were able to reproduce the results of the paper from only the data, models, code and accompanying documentation. Such collaborators would currently fall under the 'validation' role in the CRediT Taxonomy (https://casrai.org/credit/), although it should be made clear that the reproducibility co-author should not also be collaborating on the design or implementation of the analysis.

The last challenge of this section, randomness, is common in machine-learning analyses. Dataset splitting, neural network initialization and even some GPU-parallelized math used in model training all include elements of randomness. Because model outputs depend heavily on these factors, the pseudorandom number generators used in analyses must be seeded to ensure consistent results. How the seeds are set depends on the language, although authors need to take special care when working with deep-learning libraries. Current implementations often do not prioritize determinism, especially when accelerating operations on GPUs. However, some frameworks have options to mitigate nondeterministic operation (https://pytorch.org/docs/1.8.1/notes/randomness), and future versions may have fully deterministic operation (https://github.com/NVIDIA/framework-determinism). For now, the best way to account for this type of randomness is by publishing trained models. This non-determinism is another reason why the minimal standard requires model publication—reproducing the model using data and code alone may prove impossible.

As it is difficult to evaluate the extent to which an analysis follows best practices, we provide three requirements that must be met to achieve the silver standard in reproducibility. First, future users must be able to download and install all software dependencies for the analysis with a single command. Second, the order in which the analysis scripts should be run and how to run them should be documented. Finally, any random elements within the analysis should be made deterministic.

Gold

The gold standard for reproducibility requires the entire analysis to be reproducible with a single command. Achieving this goal requires authors to automate all steps of their analysis, including downloading data, pre-processing data, training models, producing output tables, and generating and annotating figures. Full automation stands in addition to tracking dependencies and making their data and code available. In short, by meeting the gold standard, authors make the burden of reproducing their work as small as possible.

Workflow management software, such as Snakemake[12] or Nextflow[13], streamline the work of meeting the gold standard. They enable authors to create a series of rules that run all the components in an analysis. While a simple shell script can also accomplish this goal, workflow management software provides several advantages without extra work from the authors. For example, workflow management software can make it easy to restart analyses after errors, parallelize analyses and track the progress of an analysis as it runs.

Caveats

Privacy. Not all data can be publicly released. Some data contain personally identifiable information or are restricted by a data use agreement. In these cases, data should be stored in a controlled access repository[14], but the use of controlled access should be explicitly approved by journals to prevent it from becoming another form of 'data available upon request'. Training models on private data also pose privacy challenges. Models trained with standard workflows can be attacked to extract training data[15]. Fortunately, model-training methods designed to preserve privacy exist: techniques such as differential privacy[16] can help make models resistant to attacks seeking to uncover personally identifiable information, and can be applied with open source libraries such as Opacus (https://opacus.ai/). Researchers working on data with privacy constraints should employ these techniques as a routine practice.

When data cannot be shared, models must be shared to have any hope of computational reproducibility. If neither data nor models are published, the code is nearly useless, as it does not have anything to operate on. Future authors could perhaps replicate the study by recollecting data and regenerating the models, but they will not be able to evaluate the original analysis based on the published materials. When working on data with privacy restrictions, it is important for authors to use privacy-preserving techniques for model training so that model release is not impeded. Studies with only models published will not be able to be fully reproduced, but there will at least be the possibility of testing model behavior on other datasets.

Compute-intensive analyses. Analyses can take a long time to run. In some cases, they may take so long to run that it is almost infeasible for them to be reproduced by a different research group.

In those cases, authors should store and publish intermediate outputs. Doing so allows other users to verify the final results even if they cannot reproduce the entire pipeline. Workflow management systems, as mentioned in the 'Gold' section, make this partial reproduction straightforward by tracking intermediate outputs and using them to reproduce the final results automatically. Setting up a lightweight analysis demonstration, such as a web app on a small dataset or a Colab notebook (https://research.google.com/colaboratory/) running a pre-trained model, can also be helpful for giving users the ability to evaluate model behavior without using large amounts of compute.

Reproducibility of packages, libraries and software products. The standards outlined in this paper focus on the computational reproducibility of analyses using machine learning. Standards for software designed for reuse, such as software packages and utilities, would have a broader scope and encompass more topics. In addition to our standards, such software should make use of unit testing, follow code style guidelines, have clear documentation[17] and ensure compatibility across major operating systems to meet the gold standard for this type of research product.

Conclusion

If we are to make machine-learning research in the life sciences trustworthy, we must make it computationally reproducible. Authors who strive to meet the bronze, silver and gold standards will increase the reproducibility of machine-learning analyses in the life sciences. These standards can also accelerate research in the field. In the status quo, there is no explicit reward for reproducible programming practices. As a result, authors can ostensibly minimize their own programming effort by using irreproducible programming practices and leaving future authors to make up the difference. In practice, irreproducible programming practices tend to decrease effort in the short term for the authors, but increase effort in the long run on both the parts of the original authors and future reproducing authors. Implementing the standards in a way that rewards reproducible science helps avoid these long-run costs (see Box 1).

Ultimately, reproducibility in computational research is often comparatively easy to experimental life sciences research. Computers are designed to perform the same tasks repeatedly with identical results. If we can not make purely computational analysis reproducible, how can we ever manage to make truly reproducible work in wet lab research with such variable factors as reagents, cell lines and environmental conditions? If we want life sciences to lead the way in trustworthy verifiable research, then setting standards for computational reproducibility is a good place to start. ❐

Benjamin J. Heil[1], Michael M. Hoffman[2,3,4,5], Florian Markowetz[6], Su-In Lee[7], Casey S. Greene[8,9] ✉ and Stephanie C. Hicks[10] ✉

[1]Genomics and Computational Biology Graduate Group, Perelman School of Medicine, University of Pennsylvania, Philadelphia, PA, USA. [2]Princess Margaret Cancer Centre, University Health Network, Toronto, Ontario, Canada. [3]Department of Medical Biophysics, University of Toronto, Toronto, Ontario, Canada. [4]Department of Computer Science, University of Toronto, Toronto, Ontario, Canada. [5]Vector Institute, Toronto, Ontario, Canada. [6]Cancer Research UK Cambridge Institute, University of Cambridge, Cambridge, UK. [7]Paul G. Allen School of Computer Science and Engineering, University of Washington, Seattle, WA, USA. [8]Department of Biochemistry and Molecular Genetics, University of Colorado School of Medicine, Aurora, CO, USA. [9]Center for Health AI, University of Colorado School of Medicine, Aurora, CO, USA. [10]Department of Biostatistics, Johns Hopkins Bloomberg School of Public Health, Baltimore, MD, USA.

✉*e-mail: casey.s.greene@cuanschutz.edu; shicks19@jhu.edu*

Published online: 30 August 2021
https://doi.org/10.1038/s41592-021-01256-7

References

1. Oreskes, N. *Why Trust Science* (Princeton Univ. Press, 2019).
2. Stodden, V., Borwein, J. & Bailey, D. H. *Comput. Sci. Res. SIAM News* **46**, 4–6 (2013).
3. Norgeot, B. et al. *Nat. Med.* **26**, 1320–1324 (2020).
4. Hernandez-Boussard, T., Bozkurt, S., Ioannidis, J. P. A. & Shah, N. H. *J. Am. Med. Inform. Assoc.* **27**, 2011–2015 (2020).
5. Mongan, J., Moy, L. & Kahn, C. E. Jr *Radiol. Artif. Intell.* **2**, e200029 (2020).
6. Wilson, S. L. et al. *FEBS Lett.* **595**, 847–863 (2021).
7. Edgar, R., Domrachev, M. & Lash, A. E. *Nucleic Acids Res.* **30**, 207–210 (2002).
8. Ellenberg, J. et al. *Nat. Methods* **15**, 849–854 (2018).
9. Avsec, Ž. et al. *Nat. Biotechnol.* **37**, 592–600 (2019).
10. Fischer, D. S. et al. Preprint at https://www.biorxiv.org/content/10.1101/2020.12.16.419036v1 (2020).
11. Merkel, D. *Linux J.* **239**, 2 (2014).
12. Köster, J. & Rahmann, S. *Bioinformatics* **28**, 2520–2522 (2012).
13. Di Tommaso, P. et al. *Nat. Biotechnol.* **35**, 316–319 (2017).
14. Byrd, J. B. et al. *Nat. Rev. Genet.* **21**, 615–629 (2020).
15. Carlini, N. et al. Preprint at https://arxiv.org/abs/2012.07805 (2020).
16. Abadi, M. et al. *Proc. ACM SIGSAC Conference on Computer and Communications Security* (eds Weippl, E. & Katzenbeisser, S.) 308–318 (ACM, 2016).
17. Karimzadeh, M. & Hoffman, M. M. *Bioinformatics* **19**, 693–699 (2018).
18. Gentleman, R. C. et al. *Genome Biol.* **5**, R80 (2004).

Acknowledgements

This work was supported by the Natural Sciences and Engineering Research Council of Canada (RGPIN-2015-03948 to M.M.H.); Cancer Research UK (A19274 to F.M.); and the National Institutes of Health's National Institute of General Medical Sciences (R35 GM128638 to S.L.), the National Human Genome Research Institute (R00HG009007 to S.C.H. and R01HG010067 to C.S.G) and the National Cancer Institute of the National Institutes of Health (R01CA237170 to C.S.G.).

Author contributions

C.S.G. was responsible for conceptualization. B.J.H. was responsible for project administration. B.J.H. and S.C.H. wrote the original draft of the manuscript; and B.J.H., S.C.H., M.M.H., S.L., F.M. and C.S.G. contributed to reviewing and editing.

Competing interests

M.M.H. has received an Nvidia GPU grant.

Additional information

Peer review information *Nature Methods* thanks Peter Koo, Brian Hie and Jianlin Cheng for their contribution to the peer review of this work.