

Reproducible Machine Learning for Humans

Nikita Kazeev on behalf on the Everware and REP teams

kazeevn@yandex-team.ru

2016-10-13, 4th National eScience Symposium, Amsterdam, the Netherlands

Yandex School of Data Analysis

<https://yandexdataschool.com>

- › A non-commercial private university
- › Free tuition, no employment obligations on part of the students (yet many go to Yandex)
- › 450+ students graduated since 2007
- › Strong (50% drop out rate) education in Data & Computer Science
- › Organizes a Machine Learning Conference
- › Interest in interdisciplinary research (eScience) — from Information Retrieval to Fundamental Science
- › 25% of our students have background in Physics
- › A full member of the LHCb experiment in CERN since 2015, an associate member during 2014-2015

Me

- › A data scientist
- › MS in Physics
- › Work on infrastructure optimization and anomaly detection for LHCb
- › Taught machine learning at Machine Learning in High Energy Physics Summer Schools

Plan

- › The problem of research irreproducibility
- › Our tools for computational experiments
 - › Everware
 - › Reproducible Experiment Platform (REP)
- › Demo

Irreproducibility indicators

- › ‘Which version of my code I used to generate figure 13?’
- › ‘The new student wants to reuse that model I published three years ago but he can’t reproduce the figures’
- › ‘I thought I’ve used the same parameters but I’m getting different results...’
- › ‘On what dataset have I compared algorithms exactly?’
- › ‘Why did I do that?!’
- › ‘It worked yesterday!!’

Case in point

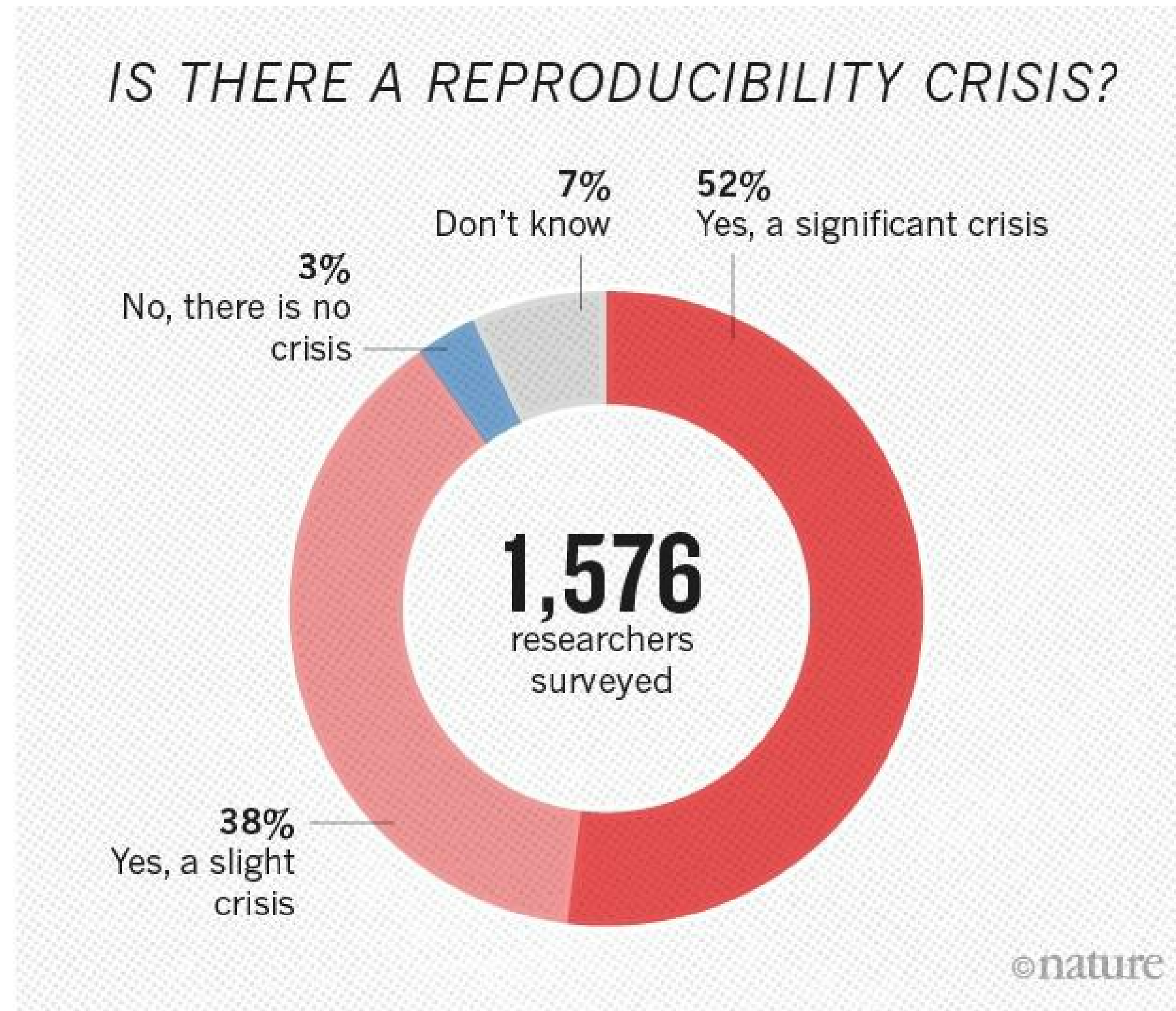
- › 53 'landmark' papers in drug discovery
- › 2012 by Amgen (US company)
- › "confirmed in only 6 (11%) cases"

- › 54 papers in cancer biology 2010-2012
- › 2013
- › US\$1.6 million
- › results, spreadsheet
- › <https://osf.io/e81xl/wiki/home/>
- › to be completed by 2017

<http://www.nature.com/nature/journal/v483/n7391/full/483531a.html>

<http://www.nature.com/news/cancer-reproducibility-project-scales-back-ambitions-1.18938>

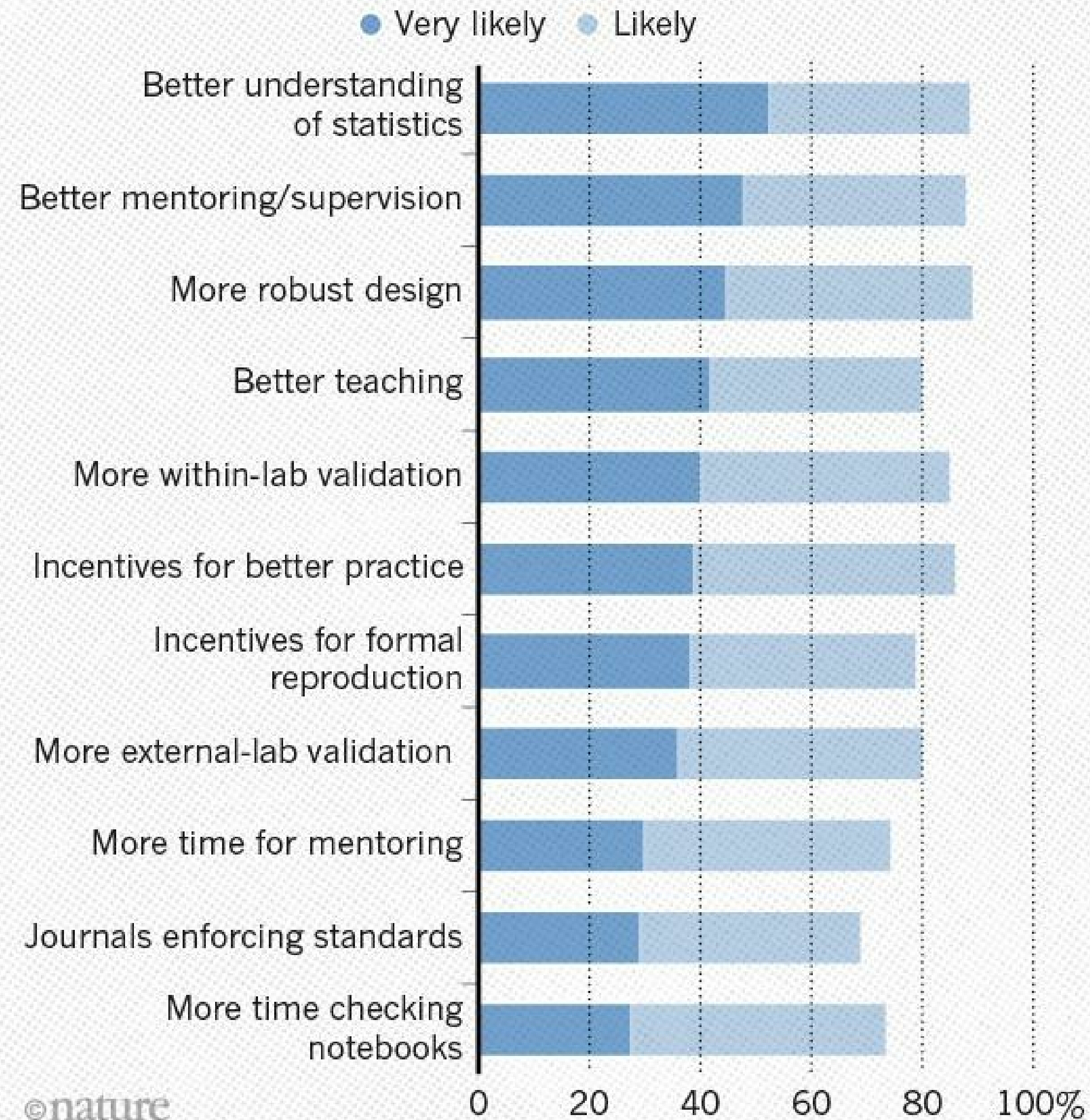
Nature's Reproducibility Survey



- › Nature: 1,500 scientists lift the lid on reproducibility by Monya Baker
- › raw survey data ([link](#))

WHAT FACTORS COULD BOOST REPRODUCIBILITY?

Respondents were positive about most proposed improvements but emphasized training in particular.



...part of the story

***Computational experiment* is a significant part of the experiment, that starts as data collected. Reproducibility of that part being just a partial answer can be aided technologically.**

Possible effects (see the previous slide):

- › Practical
 - › better mentoring/supervision
 - › more within-lab validation
 - › simplified external-lab validation
 - › incentive for better practice
 - › robust design
- › Educational
 - › wider access to the best practices
 - › better teaching

Tools in High Energy Physics

- › **data** storage
 - › shared storage (XROOTD, AFS, EOS, CERNBOX)
- › standardized **environment**
 - › software: ROOT, minuit, RooFit, experiment-stack, ...
 - › computational cluster (e.g. `lxplus`)
- › **code** versioning repository (gitlab)
- › advanced analysis approaches
 - › blind analysis
 - › reviews, cross-checks within group, inter-group collaboration
- › collaborative culture
 - › q&a groups, experts
 - › publishing workflow
- › double experiment-checks

Reproducibility key components

- › Basic assumptions (vocabulary)
- › Data
- › Environment + Resources (CPU/GPU)
- › Code
- › Workflow
- › Automated intermediate results checks
- › Final results (datasets, publications)

Common environment

Enter Reproducible Experiment Platform (**REP**)

Common environment

Enter Reproducible Experiment Platform (**REP**)

- › Python-based (numpy, pandas, ...), Jupyter-friendly

Common environment

Enter Reproducible Experiment Platform (**REP**)

- › Python-based (numpy, pandas, ...), Jupyter-friendly
- › Unified scikit-learn-like API to many ML packages (Sklearn, XGBoost, uBoost, TMVA, Theanets, ...)

Common environment

Enter Reproducible Experiment Platform (**REP**)

- › Python-based (numpy, pandas, ...), Jupyter-friendly
- › Unified scikit-learn-like API to many ML packages (Sklearn, XGBoost, uBoost, TMVA, Theanets, ...)
- › Meta-algorithms pipelines («REP-Lego»)

Common environment

Enter Reproducible Experiment Platform (**REP**)

- › Python-based (numpy, pandas, ...), Jupyter-friendly
- › Unified scikit-learn-like API to many ML packages (Sklearn, XGBoost, uBoost, TMVA, Theanets, ...)
- › Meta-algorithms pipelines («REP-Lego»)
- › Configurable interactive reporting & visualization to ensure model quality (e.g. check for overfitting)

Common environment

Enter Reproducible Experiment Platform (**REP**)

- › Python-based (numpy, pandas, ...), Jupyter-friendly
- › Unified scikit-learn-like API to many ML packages (Sklearn, XGBoost, uBoost, TMVA, Theanets, ...)
- › Meta-algorithms pipelines («REP-Lego»)
- › Configurable interactive reporting & visualization to ensure model quality (e.g. check for overfitting)
- › Pluggable quality metrics

Common environment

Enter Reproducible Experiment Platform (**REP**)

- › Python-based (numpy, pandas, ...), Jupyter-friendly
- › Unified scikit-learn-like API to many ML packages (Sklearn, XGBoost, uBoost, TMVA, Theanets, ...)
- › Meta-algorithms pipelines («REP-Lego»)
- › Configurable interactive reporting & visualization to ensure model quality (e.g. check for overfitting)
- › Pluggable quality metrics
- › Paralleled training of classifiers & grid search (IPython parallel)

Common environment

Enter Reproducible Experiment Platform (**REP**)

- › Python-based (numpy, pandas, ...), Jupyter-friendly
- › Unified scikit-learn-like API to many ML packages (Sklearn, XGBoost, uBoost, TMVA, Theanets, ...)
- › Meta-algorithms pipelines («REP-Lego»)
- › Configurable interactive reporting & visualization to ensure model quality (e.g. check for overfitting)
- › Pluggable quality metrics
- › Paralleled training of classifiers & grid search (IPython parallel)
- › Open-source, Apache 2.0: <https://github.com/yandex/rep>
- › Well-documented, supported by Yandex, <http://yandex.github.io/rep/>

A reproducible study example

<https://github.com/everware/everware-dimuon-example>

A reproducible study example

<https://github.com/everware/everware-dimuon-example>

› **data:** CERNBOX

A reproducible study example

<https://github.com/everware/everware-dimuon-example>

- › **data:** CERNBOX
- › **common environment:** REP

A reproducible study example

<https://github.com/everware/everware-dimuon-example>

- › **data:** CERNBOX
- › **common environment:** REP
- › **environment management:** Docker

A reproducible study example

<https://github.com/everware/everware-dimuon-example>

- › **data:** CERNBOX
- › **common environment:** REP
- › **environment management:** Docker
- › GitHub: analysis **code and environment versioning**

A reproducible study example

<https://github.com/everware/everware-dimuon-example>

- › **data:** CERNBOX
- › **common environment:** REP
- › **environment management:** Docker
- › GitHub: analysis **code and environment versioning**
- › continuous integration: intermediate **results checks** & report

A reproducible study example

<https://github.com/everware/everware-dimuon-example>

- › **data:** CERNBOX
- › **common environment:** REP
- › **environment management:** Docker
- › GitHub: analysis **code and environment versioning**
- › continuous integration: intermediate **results checks** & report

Steps to run:

- › install Docker
 - › <https://docs.docker.com/engine/installation/>
- › clone the repository
 - › `git clone https://github.com/everware/everware-dimuon-example.git`
- › build the Docker image (will need to download ~500 Mb)
 - › `docker build . -t dimuon`
- › run Docker with the repository folder mounted and Jupyter port forwarded
 - › `docker run -it -p 127.0.0.1:8888:8888 -v $(pwd):/notebooks dimuon bash`
- › inside run Jupyter
 - › `cd /notebooks && jupyter notebook --no-browser`
- › with the browser go to 127.0.0.1:8888

A reproducible study example

<https://github.com/everware/everware-dimuon-example>

- › **data:** CERNBOX
- › **common environment:** REP
- › **environment management:** Docker
- › GitHub: analysis **code and environment versioning**
- › continuous integration: intermediate **results checks** & report

Steps to run:

- › install Docker
 - › <https://docs.docker.com/engine/installation/>
- › clone the repository
 - › `git clone https://github.com/everware/everware-dimuon-example.git`
- › build the Docker image (will need to download ~500 Mb)
 - › `docker build . -t dimuon`
- › run Docker with the repository folder mounted and Jupyter port forwarded
 - › `docker run -it -p 127.0.0.1:8888:8888 -v $(pwd):/notebooks dimuon bash`
- › inside run Jupyter
 - › `cd /notebooks && jupyter notebook --no-browser`
- › with the browser go to 127.0.0.1:8888

A reproducible study example

<https://github.com/everware/everware-dimuon-example>

- › **data:** CERNBOX
- › **common environment:** REP
- › **environment management:** Docker
- › GitHub: analysis **code and environment versioning**
- › continuous integration: intermediate **results checks** & report

Or you can use *Everware* - just click.

Everware is ...

... about re-useable science, it allows people to jump right in to your research code. Lets you launch *Jupyter* notebooks from a git repository with a click of a button.

- › <https://github.com/everware> - Code
- › <https://everware.rep.school.yandex.net> - Yandex instance

More examples:

- › algorithm meta-analysis, https://github.com/openml/study_example
- › gravitational waves, <https://github.com/anaderi/GW150914>
- › COMET, <https://github.com/yandexdataschool/comet-example-ci>

Everware is ...

... about re-useable science, it allows people to jump right in to your research code. Lets you launch *Jupyter* notebooks from a git repository with a click of a button.

- › <https://github.com/everware> - Code
- › <https://everware.rep.school.yandex.net> - Yandex instance

More examples:

- › algorithm meta-analysis, https://github.com/openml/study_example
- › gravitational waves, <https://github.com/anaderi/GW150914>
- › COMET, <https://github.com/yandexdataschool/comet-example-ci>

Under the hood of Everware

- › an extension for *JupyterHub*:
 - › a spawner for building and running custom *Docker* images
- › integrated with:
 - › Dockerhub
 - › GitHub (for authentication and repository interaction)

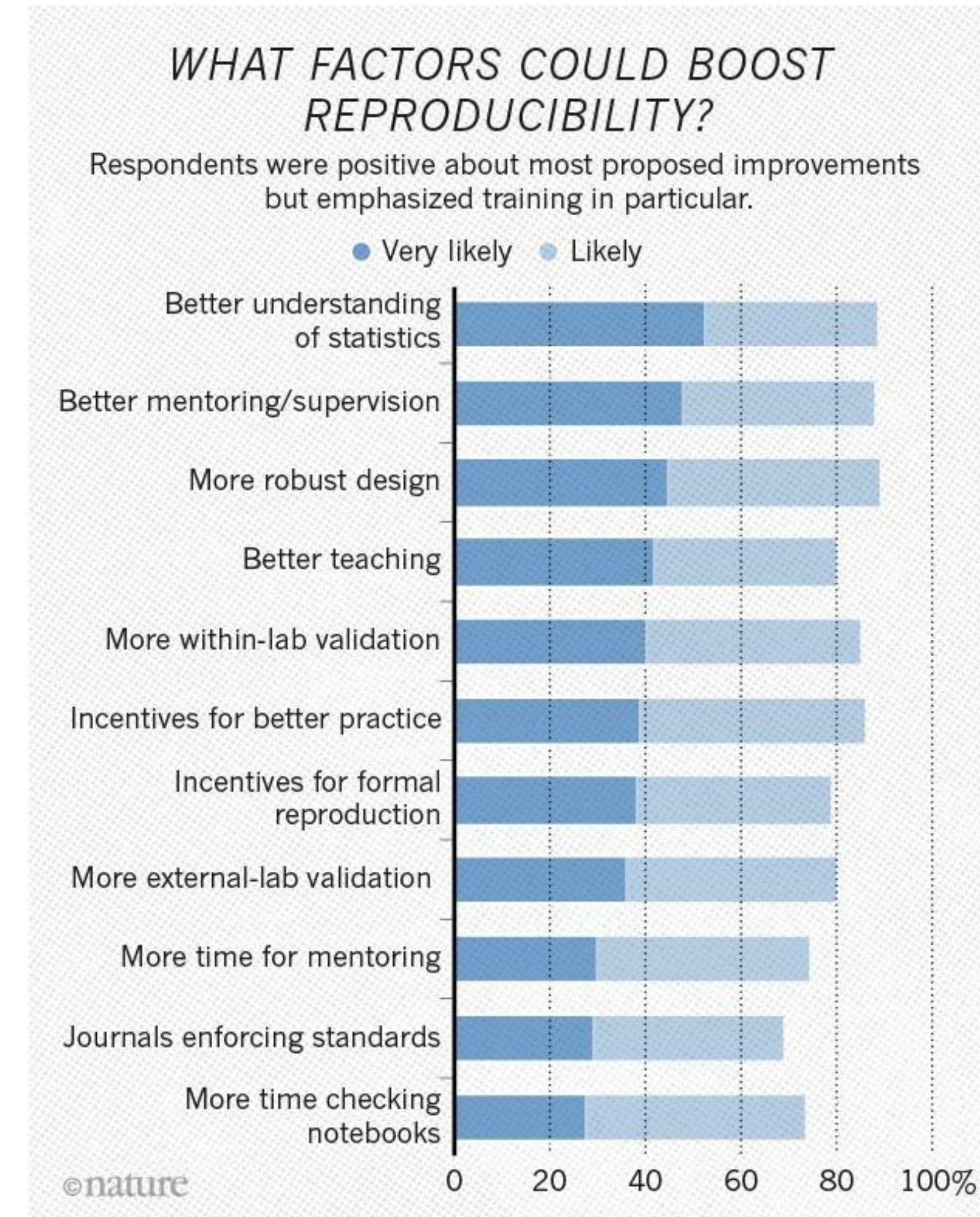
Pros & cons

Pros

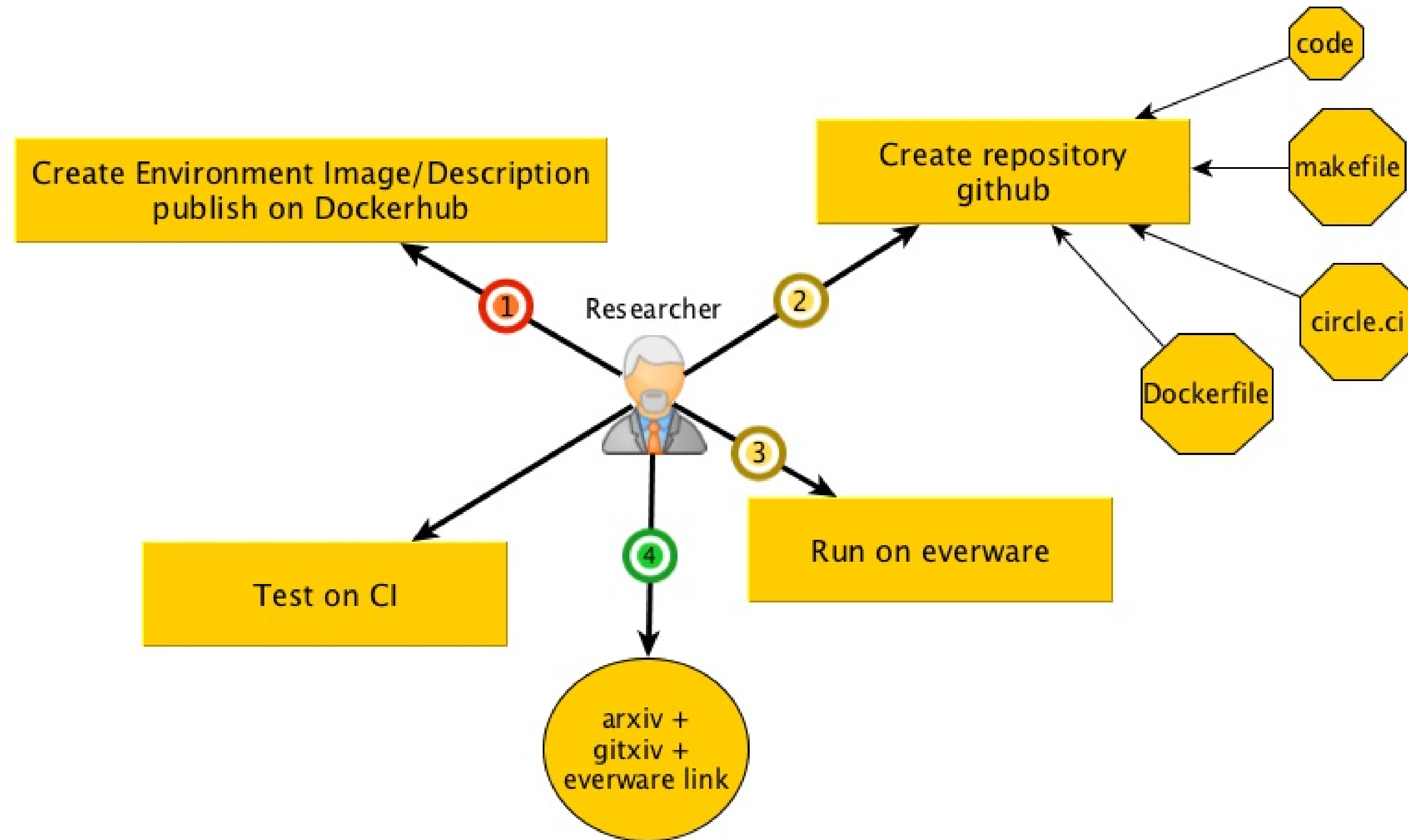
- › easier supervision/mentoring
- › easier within-lab validation
- › wider access to the best practices
- › simplified cross-lab validation
- › good incentive for formal reproduction
- › *good thing for industry career track development*
- › access to wider set of practices

Cons

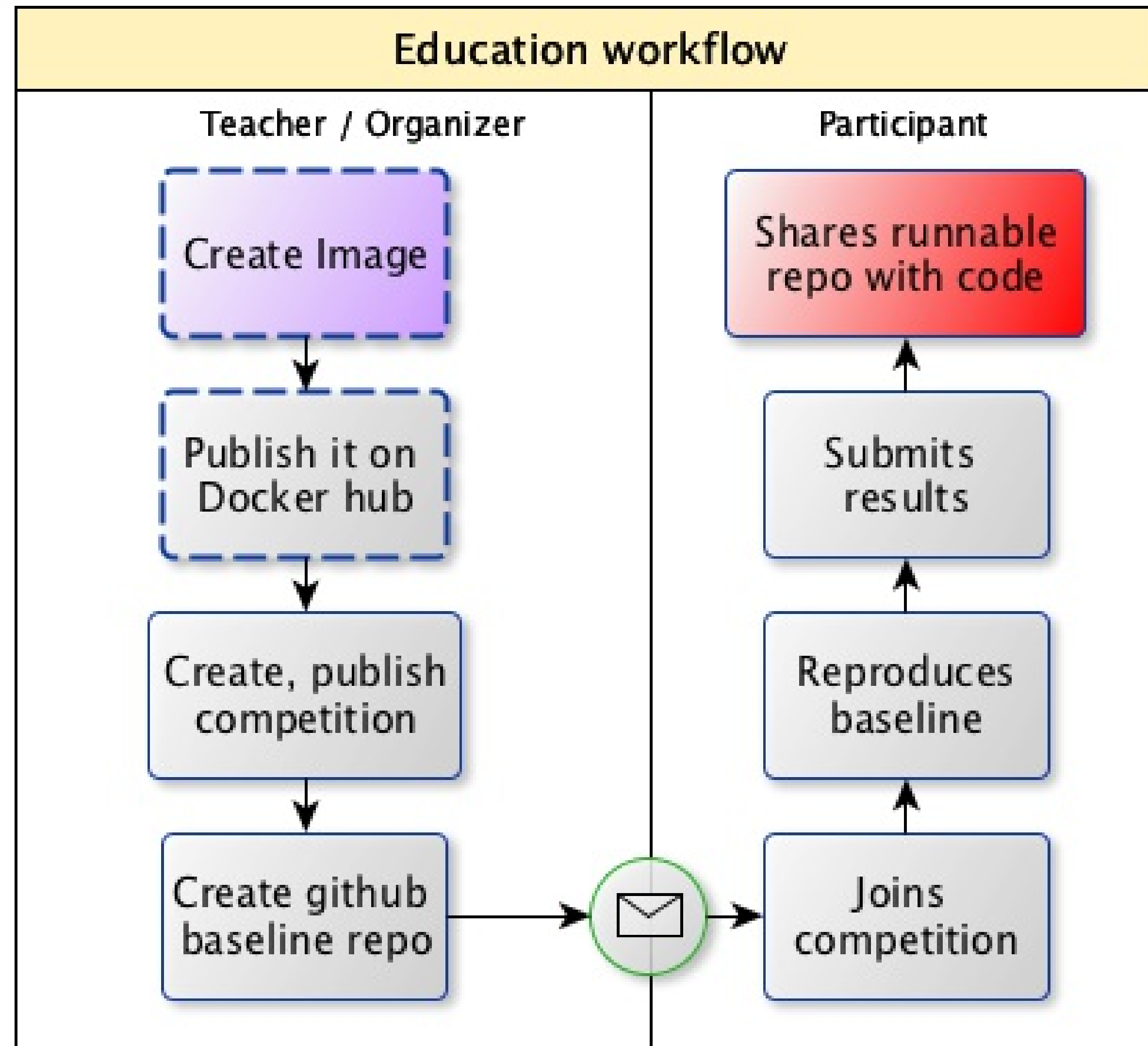
- › learning a bit of open-source technology
- › re-organize internal research process
- › inner barrier for openness
- › higher incentive for mindless *borrowing*
- › environments divergence



Research workflow with everware



Education workflow with everware



- › Python course at YSDA 2015
- › MLHEP Machine Learning summer schools 2015 and 2016
- › YSDA course on Machine learning at Imperial College London 2016
- › Kaggle competitions 2016
- › Machine learning course at University of Eindhoven
- › LHCb open data masterclass

Roadmap

- › Integrate with data sharing resources (zotero, figshare, etc)
- › Automatic capture of environment (integrate with repro-zip)
- › Integration with publishing resources (gitxiv, re-science, openml)
- › Not only jupyter-based computations
- › Bring your own resources computational model

Conclusion

- › Reproducibility depends on humans
 - › Can be helped with human-facing technology;
- › *Everware works* for research and education;
 - › easy to try;
 - › WIP, <https://github.com/everware>
 - › feature requests are welcome
 - › pull requests are most welcome
- › REP might work as a common environment for your ML study
 - › it also has nice tools to ease the routine

Thank you!

Backup

Everware demo

Running <https://github.com/everware/everware-dimuon-example>

Sorry, printed version doesn't support animation. <https://github.com/everware/everware-dimuon-example>

Yandex services landscape (est 1997)

- › Web search
- › Image search
- › Speech recognition
- › Car traffic prediction
- › Mail and spam filtering
- › Natural language translation
- › Market (shopwindow for internet shops)
- › Yandex Data Factory (<https://yandexdatafactory.com>)
- › Yandex School of Data Analysis
 - › (full member of LHCb since Dec' 15)

References

- › <http://www.nature.com/news/1-500-scientists-lift-the-lid-on-reproducibility-1.19970>
- › <https://rescience.github.io/read/>
- › <http://push.cwcon.org/>
- › <https://openml.org>
- › <https://figshare.com/>
- › <https://gitlab.cern.ch/lhcb-bandq-exotics/Lb2LcD0K>
- › <https://osf.io/ezcuj/wiki/home/>
- › <https://osf.io/e81xl/wiki/home/>
- › Center for open science, <https://cos.io/>
- › IPFS, <https://github.com/ipfs/>
- › Nature, keyword: reproducibility, <http://www.nature.com/news/reproducibility-1.17552>

Dealing with cognitive bias

