Reproducibility in Data Sciences

Why, What, and How

Dr. André Anjos (andreanjos.org)

May 3, 2021

Biosignal Processing Group

Outline

Self-Introduction

Talk

Outline

Self-Introduction

- Context
- Current WIP

Talk

Idiap Research Institute (idiap.ch)

Bio

- Independant non-profit research institute
- Founded in 1991
- Located in Martigny, Wallis – affiliation with EPFL (teaching, PhD program)
- ∼130 persons (researchers, engineers, PhD, PostDoc, ...)

14 Research groups

Themes: Speech and audio processing, computer vision and learning, Perception and activity understanding, Biometrics, Social computing, Natural language processing, Energy Informatics, Genomics, Biomedical Imaging, Robot learning and interaction, **Biosignal Procesing**.

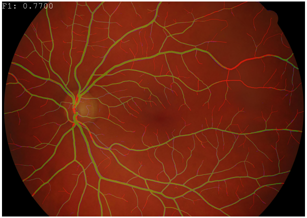
Myself

Multi-disciplinary track with AI focus:

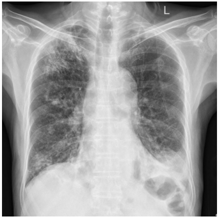
- 2001-2010 Computer vision and Machine Learning for High-Energy Physics (Ph.D and postdoc)
- 2010-2018 Biometrics (Research Associate)
- 2018- Biosignals (Researcher):
  - Applications to medical and health data
  - Computer vision, Machine learning and Sequence processing
  - Reproducibility and "Platforms"
  - Teaching

More information: **http://andreanjos.org**

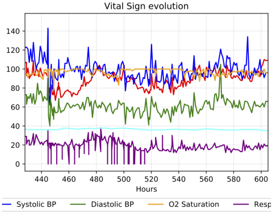
Current Work: http://andreanjos.org/research/



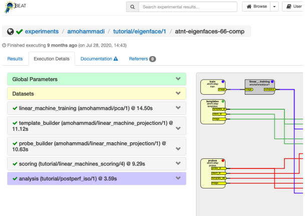
Segmentation (Retinography)



CAD (PTB from CXR)



Sequence Analysis (Vital Signs)



experiments / amohammadi / tutorial/eigenface/1 / atnt-eigenfaces-66-comp

Finished executing **9 months ago** (on Jul 28, 2020, 14:43)

Results | Execution Details | Documentation | Referrers 0

- Global Parameters
- Datasets
- ✓ linear_machine_training (amohammadi/pca/1) @ 14.50s
- ✓ template_builder (amohammadi/linear_machine_projection/1) @ 11.12s
- ✓ probe_builder (amohammadi/linear_machine_projection/1) @ 10.63s
- ✓ scoring (tutorial/linear_machines_scoring/4) @ 9.29s
- ✓ analysis (tutorial/postperf_iso/1) @ 3.59s

Reproducibility and "Platforms"

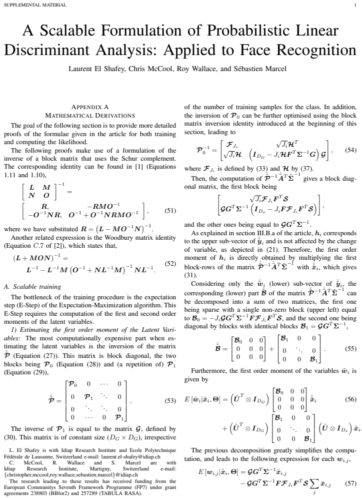
Outline

Self-Introduction

Talk

- Why?
- What?
- How?
- Conclusions

Reproducing a Paper



SUPPLEMENTAL MATERIAL 1

A Scalable Formulation of Probabilistic Linear Discriminant Analysis: Applied to Face Recognition

Laurent El Shafey, Chris McCool, Roy Wallace, and Sébastien Marcel

Appendix A
Mathematical Derivations

The goal of the following section is to provide more detailed proofs of the formulae given in the article for both training and computing the likelihood.

The following proofs make use of a formulation of the inverse of a block matrix that uses the Schur complement. The corresponding identity can be found in [1] (Equations 1.11 and 1.10),

$$\begin{bmatrix} \boldsymbol{L} & \boldsymbol{M} \\ \boldsymbol{N} & \boldsymbol{O} \end{bmatrix}^{-1} = \begin{bmatrix} \boldsymbol{R}, & -\boldsymbol{R}\boldsymbol{M}\boldsymbol{O}^{-1} \\ -\boldsymbol{O}^{-1}\boldsymbol{N}\boldsymbol{R}, & \boldsymbol{O}^{-1}+\boldsymbol{O}^{-1}\boldsymbol{N}\boldsymbol{R}\boldsymbol{M}\boldsymbol{O}^{-1} \end{bmatrix}, \quad (51)$$

where we have substituted $\boldsymbol{R} = \left(\boldsymbol{L} - \boldsymbol{M}\boldsymbol{O}^{-1}\boldsymbol{N}\right)^{-1}$.

Another related expression is the Woodbury matrix identity (Equation C.7 of [2]), which states that,

$$\left(\boldsymbol{L}+\boldsymbol{M}\boldsymbol{O}\boldsymbol{N}\right)^{-1} = \boldsymbol{L}^{-1} - \boldsymbol{L}^{-1}\boldsymbol{M}\left(\boldsymbol{O}^{-1}+\boldsymbol{N}\boldsymbol{L}^{-1}\boldsymbol{M}\right)^{-1}\boldsymbol{N}\boldsymbol{L}^{-1}. \quad (52)$$

A. Scalable training

The bottleneck of the training procedure is the expectation step (E-Step) of the Expectation-Maximization algorithm. This E-Step requires the computation of the first and second order moments of the latent variables.

1) Estimating the first order moment of the Latent Variables: The most computationally expensive part when estimating the latent variables is the inversion of the matrix $\mathring{\bar{\boldsymbol{\mathcal{P}}}}$ (Equation (27)). This matrix is block diagonal, the two blocks being $\boldsymbol{\mathcal{P}}_0$ (Equation (28)) and (a repetition of) $\boldsymbol{\mathcal{P}}_1$ (Equation (29)),

$$\mathring{\bar{\boldsymbol{\mathcal{P}}}} = \begin{bmatrix} \boldsymbol{\mathcal{P}}_0 & 0 & \cdots & 0 \\ 0 & \boldsymbol{\mathcal{P}}_1 & \ddots & 0 \\ 0 & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & \boldsymbol{\mathcal{P}}_1 \end{bmatrix}. \quad (53)$$

The inverse of $\boldsymbol{\mathcal{P}}_1$ is equal to the matrix $\boldsymbol{\mathcal{G}}$, defined by (30). This matrix is of constant size $(D_G \times D_G)$, irrespective of the number of training samples for the class. In addition, the inversion of $\boldsymbol{\mathcal{P}}_0$ can be further optimised using the block matrix inversion identity introduced at the beginning of this section, leading to

$$\boldsymbol{\mathcal{P}}_0^{-1} = \begin{bmatrix} \boldsymbol{\mathcal{F}}_{J_i} & \sqrt{J_i}\boldsymbol{\mathcal{H}}^T \\ \sqrt{J_i}\boldsymbol{\mathcal{H}} & \left(\boldsymbol{I}_{D_G} - J_i\boldsymbol{\mathcal{H}}\boldsymbol{F}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{G}\right)\boldsymbol{\mathcal{G}} \end{bmatrix}, \quad (54)$$

where $\boldsymbol{\mathcal{F}}_{J_i}$ is defined by (33) and $\boldsymbol{\mathcal{H}}$ by (37).

Then, the computation of $\mathring{\bar{\boldsymbol{\mathcal{P}}}}^{-1}\mathring{\bar{\boldsymbol{A}}}^T\bar{\boldsymbol{\Sigma}}^{-1}$ gives a block diagonal matrix, the first block being

$$\begin{bmatrix} \sqrt{J_i}\boldsymbol{\mathcal{F}}_{J_i}\boldsymbol{F}^T\boldsymbol{\mathcal{S}} \\ \boldsymbol{\mathcal{G}}\boldsymbol{G}^T\boldsymbol{\Sigma}^{-1}\left(\boldsymbol{I}_{D_x} - J_i\boldsymbol{F}\boldsymbol{\mathcal{F}}_{J_i}\boldsymbol{F}^T\boldsymbol{\mathcal{S}}\right) \end{bmatrix},$$

and the other ones being equal to $\boldsymbol{\mathcal{G}}\boldsymbol{G}^T\boldsymbol{\Sigma}^{-1}$.

As explained in section III.B.a of the article, $\boldsymbol{h}_i$ corresponds to the upper sub-vector of $\mathring{\bar{\boldsymbol{y}}}_i$ and is not affected by the change of variable, as depicted in (21). Therefore, the first order moment of $\boldsymbol{h}_i$ is directly obtained by multiplying the first block-rows of the matrix $\mathring{\bar{\boldsymbol{\mathcal{P}}}}^{-1}\mathring{\bar{\boldsymbol{A}}}^T\bar{\boldsymbol{\Sigma}}^{-1}$ with $\mathring{\bar{\boldsymbol{x}}}_i$, which gives (31).

Considering only the $\mathring{\bar{\boldsymbol{w}}}_i$ (lower) sub-vector of $\mathring{\bar{\boldsymbol{y}}}_i$, the corresponding (lower) part $\mathring{\bar{\boldsymbol{\mathcal{B}}}}$ of the matrix $\mathring{\bar{\boldsymbol{\mathcal{P}}}}^{-1}\mathring{\bar{\boldsymbol{A}}}^T\bar{\boldsymbol{\Sigma}}^{-1}$ can be decomposed into a sum of two matrices, the first one being sparse with a single non-zero block (upper left) equal to $\boldsymbol{\mathcal{B}}_0 = -J_i\boldsymbol{\mathcal{G}}\boldsymbol{G}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{F}\boldsymbol{\mathcal{F}}_{J_i}\boldsymbol{F}^T\boldsymbol{\mathcal{S}}$, and the second one being diagonal by blocks with identical blocks $\boldsymbol{\mathcal{B}}_1 = \boldsymbol{\mathcal{G}}\boldsymbol{G}^T\boldsymbol{\Sigma}^{-1}$,

$$\mathring{\bar{\boldsymbol{\mathcal{B}}}} = \begin{bmatrix} \boldsymbol{\mathcal{B}}_0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} + \begin{bmatrix} \boldsymbol{\mathcal{B}}_1 & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & \boldsymbol{\mathcal{B}}_1 \end{bmatrix}. \quad (55)$$

Furthermore, the first order moment of the variables $\bar{\boldsymbol{w}}_i$ is given by

$$E\left[\bar{\boldsymbol{w}}_i|\bar{\boldsymbol{x}}_i,\boldsymbol{\Theta}\right] = \left(\bar{\boldsymbol{U}}^T \otimes \boldsymbol{I}_{D_G}\right)\begin{bmatrix} \boldsymbol{\mathcal{B}}_0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}\mathring{\bar{\boldsymbol{x}}}_i + \left(\bar{\boldsymbol{U}}^T \otimes \boldsymbol{I}_{D_G}\right)\begin{bmatrix} \boldsymbol{\mathcal{B}}_1 & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & \boldsymbol{\mathcal{B}}_1 \end{bmatrix}\left(\bar{\boldsymbol{U}} \otimes \boldsymbol{I}_{D_x}\right)\bar{\boldsymbol{x}}_i. \quad (56)$$

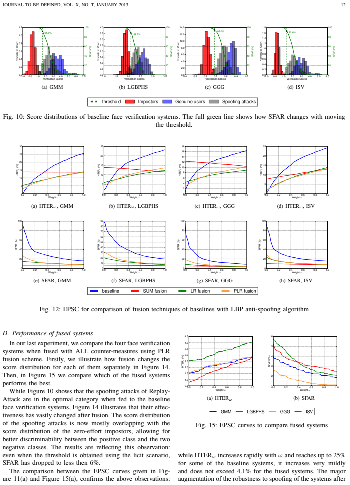
The previous decomposition greatly simplifies the computation, and leads to the following expression for each $\boldsymbol{w}_{i,j}$,

$$E\left[\boldsymbol{w}_{i,j}|\bar{\boldsymbol{x}}_i,\boldsymbol{\Theta}\right] = \boldsymbol{\mathcal{G}}\boldsymbol{G}^T\boldsymbol{\Sigma}^{-1}\bar{\boldsymbol{x}}_{i,j} - \boldsymbol{\mathcal{G}}\boldsymbol{G}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{F}\boldsymbol{\mathcal{F}}_{J_i}\boldsymbol{F}^T\boldsymbol{\mathcal{S}}\sum_j \bar{\boldsymbol{x}}_{i,j} \quad (57)$$

L. El Shafey is with Idiap Research Institute and Ecole Polytechnique Fédérale de Lausanne, Switzerland e-mail: laurent.el-shafey@idiap.ch

C. McCool, R. Wallace and S. Marcel are with Idiap Research Institute, Martigny, Switzerland e-mail: {christopher.mccool,roy.wallace,sebastien.marcel}@idiap.ch

The research leading to these results has received funding from the European Communitys Seventh Framework Programme (FP7) under grant agreements 238803 (BBfor2) and 257289 (TABULA RASA).



JOURNAL TO BE DEFINED, VOL. X, NO. T, JANUARY 2013 12

Fig. 10: Score distributions of baseline face verification systems. The full green line shows how SFAR changes with moving the threshold.

Fig. 12: EPSC for comparison of fusion techniques of baselines with LBP anti-spoofing algorithm

D. Performance of fused systems

In our last experiment, we compare the four face verification systems when fused with ALL counter-measures using PLR fusion scheme. Firstly, we illustrate how fusion changes the score distribution for each of them separately in Figure 14. Then, in Figure 15 we compare which of the fused systems performs the best.

While Figure 10 shows that the spoofing attacks of Replay-Attack are in the optimal category when fed to the baseline face verification systems, Figure 14 illustrates that their effectiveness has vastly changed after fusion. The score distribution of the spoofing attacks is now mostly overlapping with the score distribution of the zero-effort impostors, allowing for better discriminability between the positive class and the two negative classes. The results are reflecting this observation: even when the threshold is obtained using the licit scenario, SFAR has dropped to less then 6%.

The comparison between the EPSC curves given in Figure 11(a) and Figure 15(a), confirms the above observations: while HTER$_\omega$ increases rapidly with ω and reaches up to 25% for some of the baseline systems, it increases very mildly and does not exceed 4.1% for the fused systems. The major augmentation of the robustness to spoofing of the systems after

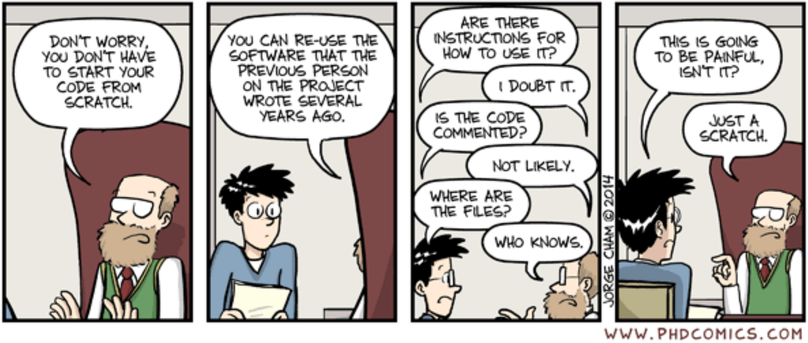
Fig. 15: EPSC curves to compare fused systems

Reproducing a Third-Party Article

You found a nice paper – work day and night to **incorporate some results** on your own project but:

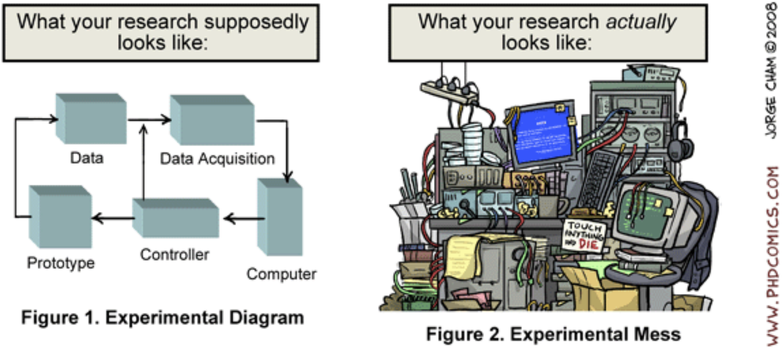
- There were **untold parameters** that needed adjustment and you couldn't get hold of them
- You realized the proposed solution **worked only on the specific data** shown at the original paper
- You realized that something did **not quite add up** in the end

Reproducing a Colleague's Results


DON'T WORRY, YOU DON'T HAVE TO START YOUR CODE FROM SCRATCH.
YOU CAN RE-USE THE SOFTWARE THAT THE PREVIOUS PERSON ON THE PROJECT WROTE SEVERAL YEARS AGO.
ARE THERE INSTRUCTIONS FOR HOW TO USE IT?
I DOUBT IT.
IS THE CODE COMMENTED?
NOT LIKELY.
WHERE ARE THE FILES?
WHO KNOWS.
JORGE CHAM © 2014
THIS IS GOING TO BE PAINFUL, ISN'T IT?
JUST A SCRATCH.
WWW.PHDCOMICS.COM

*Had **to take over** the work from another colleague that left and had to start from scratch - months into programming to make things work again*

Reproducing Your Own Work

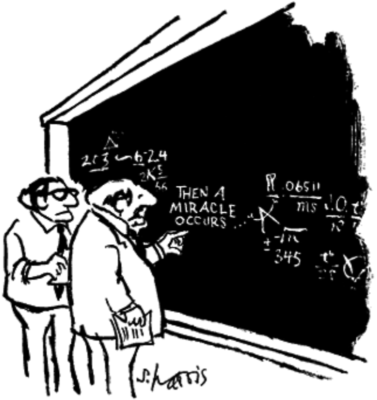

What your research supposedly looks like:
Data
Data Acquisition
Prototype
Controller
Computer
Figure 1. Experimental Diagram
What your research *actually* looks like:
TOUCH ANYTHING AND DIE
Figure 2. Experimental Mess
JORGE CHAM © 2008
WWW.PHDCOMICS.COM

Would have liked to ***replay to someone about your work****, but you couldn't really remember all details when you first made it work? Or you* ***could not make it work*** *at all?*

Poll: Have you faced this before?

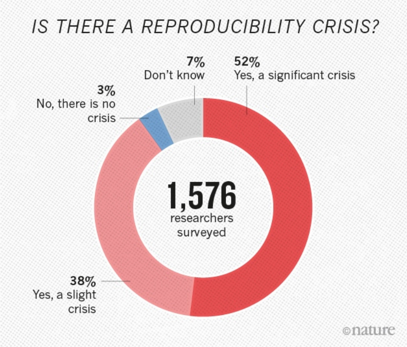
Poll: Have you face a similar situation? Mark all that apply:

- □ Irreproducible results because of missing information
- □ Irreproducible because of errors in the hypothesis, data or code
- □ Irreproducible because colleague left you with a messy code base
- □ Irreproducible because you cannot remember how to run your own experiments



"I think you should be more explicit here in step two."

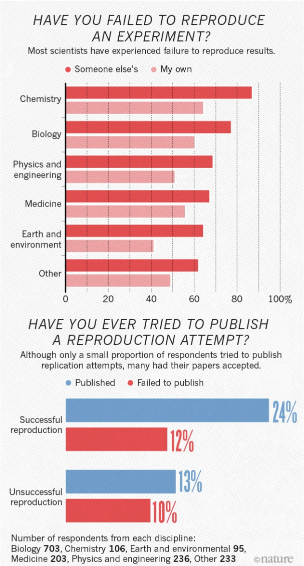
Is there a crisis?[1]



A survey in Nature revealed that irreproducible experiments are a problem across all domains of science.

[1] 1,500 scientists lift the lid on reproducibility, Monya Baker, Nature, 2016

Who else is affected?



HAVE YOU FAILED TO REPRODUCE AN EXPERIMENT?

Most scientists have experienced failure to reproduce results.

HAVE YOU EVER TRIED TO PUBLISH A REPRODUCTION ATTEMPT?

Although only a small proportion of respondents tried to publish replication attempts, many had their papers accepted.

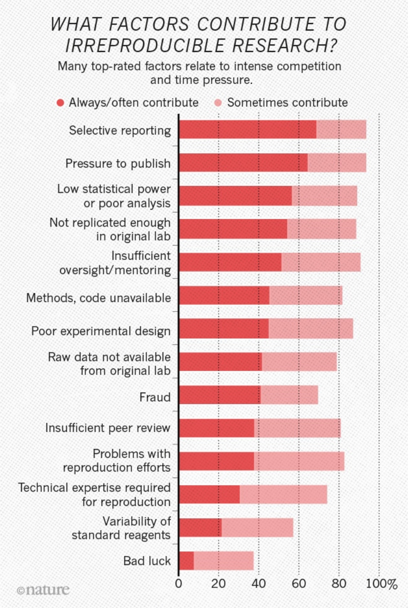
Number of respondents from each discipline: Biology **703**, Chemistry **106**, Earth and environmental **95**, Medicine **203**, Physics and engineering **236**, Other **233**

Engineering is among the most affected research fields.

For example, another Nature report[2] found that 47 out of 53 medical research papers focused on cancer research were irreproducible.

[2] Begley, C., Ellis, L. Raise standards for preclinical cancer research. Nature 483, 531–533 (2012). **https://doi.org/10.1038/483531a**

Reasons


WHAT FACTORS CONTRIBUTE TO IRREPRODUCIBLE RESEARCH?
Many top-rated factors relate to intense competition and time pressure.
● Always/often contribute
● Sometimes contribute
Selective reporting
Pressure to publish
Low statistical power or poor analysis
Not replicated enough in original lab
Insufficient oversight/mentoring
Methods, code unavailable
Poor experimental design
Raw data not available from original lab
Fraud
Insufficient peer review
Problems with reproduction efforts
Technical expertise required for reproduction
Variability of standard reagents
Bad luck
0
20
40
60
80
100%
©nature

An early advocate

> *An article about computational science in a scientific publication is **not the scholarship** itself, it is merely **advertising** of the scholarship. The actual scholarship is the complete software development environment and the complete set of instructions which generated the figures.*
>
> *D. Donoho, 2010*[3]

[3] **http://biostatistics.oxfordjournals.org/content/11/3/385.long**

Enter “Reproducible Research" (RR)[4]

One term that aggregates work comprising of:

- a **report**, that describe your work in all relevant details
- **code** to reproduce all results
- **data** required to reproduce the results
- **instructions**, on how to apply the *code* on the *data* to repeat the results on the *report*.

[4] **http://reproducibleresearch.net**

Make the difference[5]

		Data	
		Same	Different
Code	Same	**Reproducible**	Replicable
	Different	Robust	Generalisable

[5] *Identifying and Overcoming Threats to Reproducibility, Replicability, Robustness, and Generalizability in Microbiome Research*, Patrick D. Schloss, 2018

It’s all relative

Notice reproducibility is relative to the peers (not the provider).

It is reproducible

If the peer has access to data, software **and** instructions.
E.g.: AI solution is distributed by Alice, but data is available publicly.

It’s all relative

Notice reproducibility is relative to the peers (not the provider).

It is reproducible

If the peer has access to data, software **and** instructions.
E.g.: AI solution is distributed by Alice, but data is available publicly.

It is NOT reproducible

If the peer does **NOT** have access to data, software, **or** the instructions to repeat the findings.
E.g.: AI solution is distributed by John. Data is kept private or is inaccessible to Fernando.

It's all relative

Notice reproducibility is relative to the peers (not the provider).

It is reproducible

If the peer has access to data, software **and** instructions.
E.g.: AI solution is distributed by Alice, but data is available publicly.

It is NOT reproducible

If the peer does **NOT** have access to data, software, **or** the instructions to repeat the findings.
E.g.: AI solution is distributed by John. Data is kept private or is inaccessible to Fernando.

Open-sourcing

Notice that "open-sourcing" your project does not necessarily make it more reproducible. It may improve your impact though!

Levels of Reproducibility[6]

With respect to an independent researcher (reader):

0. Irreproducible
1. Cannot seem to reproduce
2. Reproducible, with extreme effort (> 1 month)
3. Reproducible, with considerable effort (> 1 week)
4. Easily reproducible ($\sim$ 15 min.), but requires proprietary software (e.g. Matlab)
5. **Easily reproducible ($\sim$ 15 min.), only free software**

[6] *Reproducible Research in Signal Processing: What, why and how*, Vandewalle, Kovacevic and Vetterli, 2012

Fields of Application

Requirements

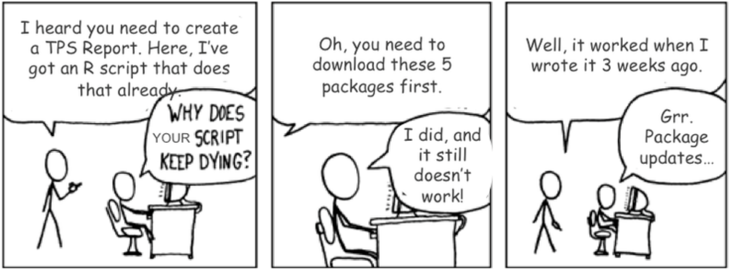
- Data that serves as input must be copiable
- Procedure must be easily copied:
  - Computer-based routines
  - Statistical or Deterministic methods

Counter-Examples

- Theoretical Physics (or disciplines of any sort)
- Biological Experimentation (see “replicability")
- Humanities
- . . .

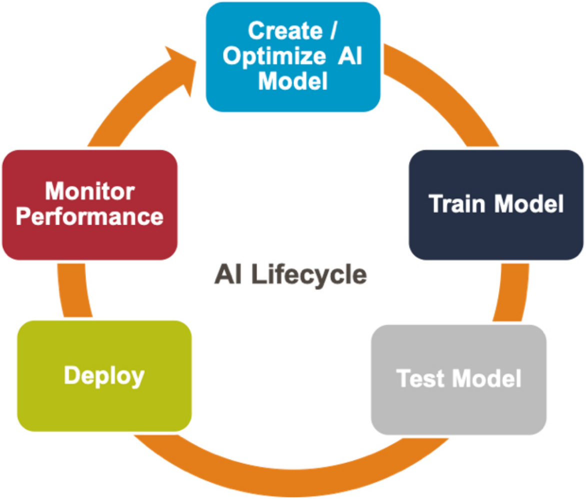
Continual Reproducibility

It is rarely the case your system is reproduced on the exact day you make it available.


I heard you need to create a TPS Report. Here, I've got an R script that does that already.
WHY DOES YOUR SCRIPT KEEP DYING?
Oh, you need to download these 5 packages first.
I did, and it still doesn't work!
Well, it worked when I wrote it 3 weeks ago.
Grr. Package updates...

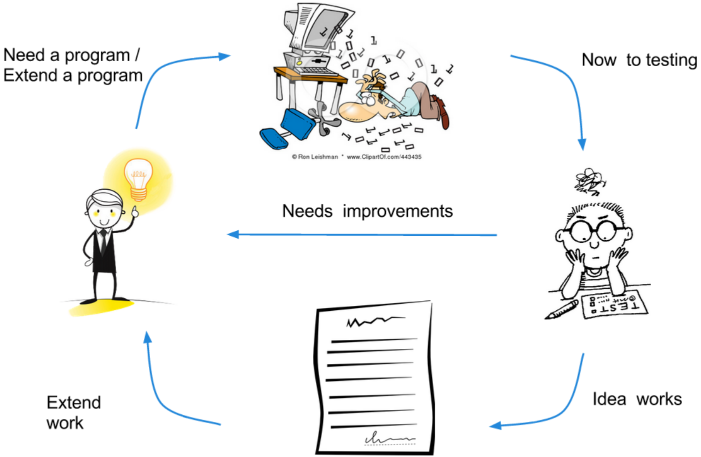
Lifecycle of AI projects: "Incrementalist"

More commonly, the lifecycle of AI projects is a continuous optimization loop



Continual or Full Reproducibility

It does **not** make sense to make a single time effort for reproducibility in AI. It is more sensible to think of it as an iterative process in search for better solutions. This is typically called **Continual** or **Full** Reproducibility.



Need a program / Extend a program

Now to testing

Needs improvements

Idea works

Extend work

Key Elements

These are important elements in Continual Reproducibility[7][8]:

- Data Management
- Framework Organization
- Version Control
- Code Sharing Platform
- Unit testing and Continuous Integration
- Documentation
- Packaging and Deployment

[7] Lack of these may affect long-term reproducibility, but do **not** deny it.

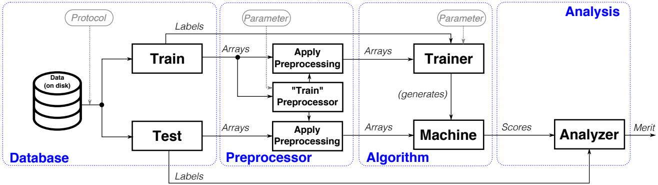
[8] *Continuously Reproducing Toolchains in Pattern Recognition and Machine Learning Experiments*, A. Anjos and others, 2017

Data Management

- **Keep** raw data.
  - Meaningful data is precious
  - Ensure you can always go back to the source
  - Keep backups
- **Make** data machine readable.
  - Use meaningful names for your variables
  - Avoid proprietary formats (to improve long term storage)
- **Record** all the steps used to produce and process data
  - Do not improvise, script everything
  - Write down documentation for your data that is meaningful to you and collaborators
- **Distribute** the data (if open-access)
  - Through DOI-capable portal (e.g. Zenodo)
  - Disclaimer: More “research" oriented

Framework Organization

Encapsulate components you would like to test



- Data and evaluation protocols must be recorded and provided like simple iterators
- Preprocessing steps
- Your Machine Learning solution
- How to analyze the data
- Encapsulate components for easily replacing each upon need

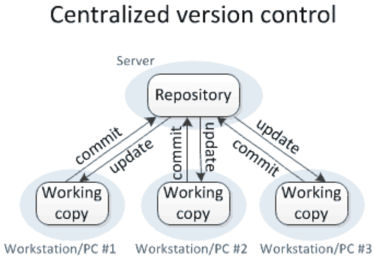
Framework Execution: helping your peers

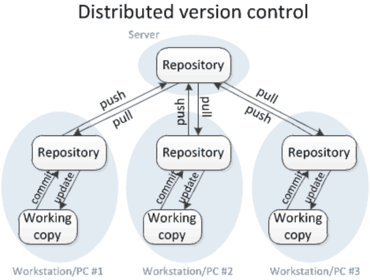
Very often, executing your experiments is not that simple (requires multiple coordinated steps). Here are some workflow managers you may consider then:

- Dask (**https://dask.org**) - widely used throughout AI community for scaling-up workflows, backed by various actors in the Python ecosystem
- Snakemake (**https://bitbucket.org/snakemake/snakemake/**) - used in niche areas, support for conda and docker-based execution
- All you can eat about workflow management: **https://github.com/pditommaso/awesome-pipeline**
- New ones being added everyday. Try to search for "workflow" or "pipeline" management in your preferred programming language.

What is Revision Control?

- Management of changes to documents/code or any sorts of collections of information
- It is normally done by specialized software packages such as `git`
- There are two types:
  - Centralized: Revision history is kept on a remote server
  - Distributed: History is copied with the repository





Why is it necessary?

Imagine a world w/o version control:

- You released version 1.0 of your software. It has a bug. Which other versions are affected?
- When was the last time I touched this file? Which changes did I do?
- You introduced a bug on the software: Where is that *fracking* backup?

It is possible!

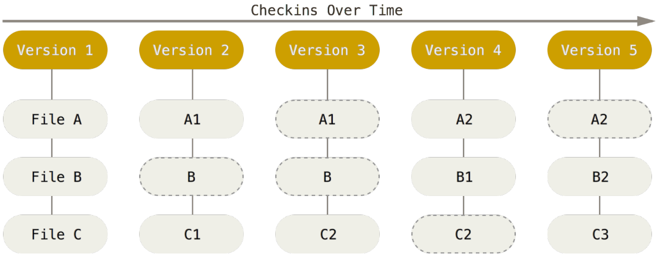
Actually, the Linux project stayed 11 years w/o version control!

This was possible thanks to an “extremely" organized procedure for diff/patching changes that gave birth to what is “Git" today!

De facto standard: Git

What is Git?

Git is a distributed revision control system. It keeps snapshots of **the entirety** of your versioned directory through time using patches.



Checkins Over Time

Version 1: File A, File B, File C

Version 2: A1, B, C1

Version 3: A1, B, C2

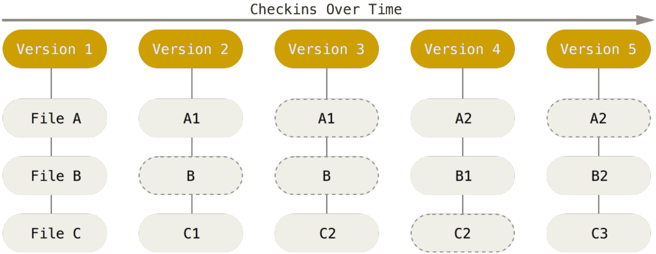
Version 4: A2, B1, C2

Version 5: A2, B2, C3

De facto standard: Git

What is Git?

Git is a distributed revision control system. It keeps snapshots of **the entirety** of your versioned directory through time using patches.

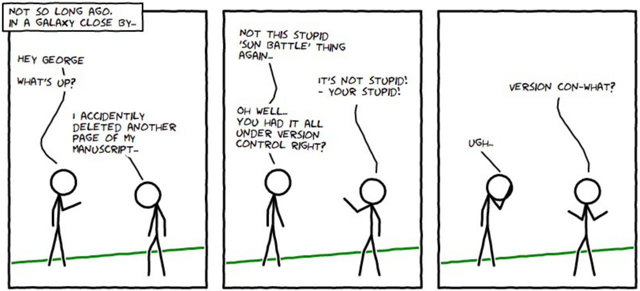


Old tools, new usage

In order to create a snapshot, git uses *diffs*, *patches* and (SHA-1) *hashes*

Version Control

Tip: Version control everything!


NOT SO LONG AGO. IN A GALAXY CLOSE BY...
HEY GEORGE
WHAT'S UP?
I ACCIDENTILY DELETED ANOTHER PAGE OF MY MANUSCRIPT...
NOT THIS STUPID 'SUN BATTLE' THING AGAIN...
OH WELL... YOU HAD IT ALL UNDER VERSION CONTROL RIGHT?
IT'S NOT STUPID! - YOUR STUPID!
UGH...
VERSION CON-WHAT?

- As a consequence, prefer text files to other types of input
  - Programs are good candidates
  - Document your code using simple mark-up
  - Use LaTeX or mark-up for your reports
  - Microsoft `<name>` is not a good candidate...

Code Sharing

Keep track of issues, annotate recipes

- There are several alternatives in the free-world: GitHub, GitLab, Bitbucket, etc. Just pick one! Key advantages:
  - Free for open-source projects
  - Private repositories for academic usage
  - Allows easy code sharing and free web-hosting for your projects
  - Integrates wiki so you can setup various sorts of guides
  - Integrates an issue tracker so people can report bugs and you can keep track of development
  - Allows the creation of a website for your project for free
- If maintaining an in-house solution, make it backup regularly
- Interact with your code-sharing repository often to avoid local, unbacked-up, changes
- Use it as a portal during reports to your management and to keep track of decisions

Unit testing and Continous Integration

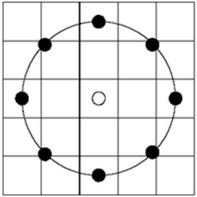
Test every piece of code you can

- Do **NOT** think you are not subject to errors
  - At first, the code is probably going to be OK
  - As the project develops, you will add code and skip checking basics - that is when errors tend to appear
  - Having a thorough check you can run under a minute helps **a lot**
- This is probably the hallmark of **reliable** code
- There is no *right* amount of testing. It should be thorough and pertinent
- Use your code sharing solution to run tests for you at every push, so you can **trust** your own changes
- If you collaborate (internally or externally), this also ensures your colleagues' additions do not break your analysis!
- Underlying libraries and code may also contain unexpected behaviour!

Unit testing: A Case Study

Local Binary Patterns, $>$**15k citations** - Matlab Toolbox (UOULU)[9]

$\mathrm{LBP}_{8,r2}$



Evaluate on position

Compare

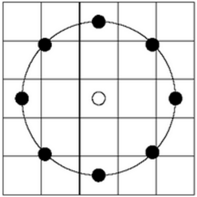
Get a code

[9] *Multiresolution gray-scale and rotation invariant texture classification with local binary patterns*, Ojala, 2002 (TPAMI)

Unit testing: A Case Study

Local Binary Patterns, >**15k citations** - Matlab Toolbox (UOULU)[9]

$\mathrm{LBP}_{8,r2}$



Bug at Matlab bi-linear interpolation

Evaluate on position

Interpolation Error

Compare

Error Propagated

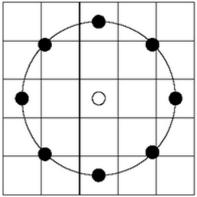
Get a code

[9] *Multiresolution gray-scale and rotation invariant texture classification with local binary patterns*, Ojala, 2002 (TPAMI)

Unit testing: A Case Study

Local Binary Patterns, >**15k citations** - Matlab Toolbox (UOULU)[9]

LBP$_{8,r2}$



Bug at Matlab bi-linear interpolation

Evaluate on position

Interpolation Error

Compare

Error Propagated

Get a code

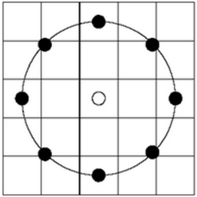
In natural images, differences can amount to ∼5% of codes

[9] *Multiresolution gray-scale and rotation invariant texture classification with local binary patterns*, Ojala, 2002 (TPAMI)

Unit testing: A Case Study

Local Binary Patterns, >**15k citations** - Matlab Toolbox (UOULU)[9]

$\mathrm{LBP}_{8,r2}$



Bug at Matlab bi-linear interpolation

Evaluate on position

Interpolation Error

Compare

Error Propagated

Get a code

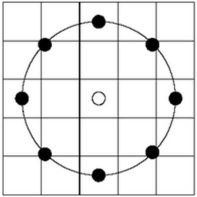
How to plan for errors?

[9] *Multiresolution gray-scale and rotation invariant texture classification with local binary patterns*, Ojala, 2002 (TPAMI)

Unit testing: A Case Study

Local Binary Patterns, >**15k citations** - Matlab Toolbox (UOULU)[9]

$\mathrm{LBP}_{8,r2}$



Bug at Matlab bi-linear interpolation

Evaluate on position

Interpolation Error

Compare

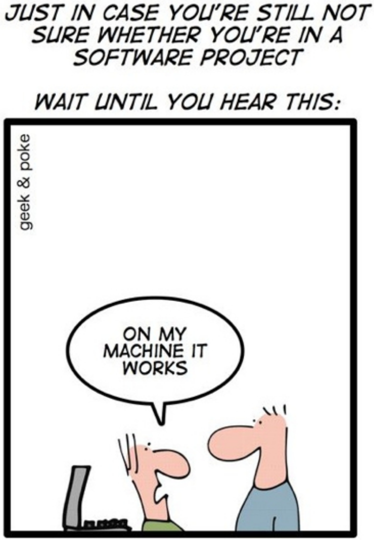
Error Propagated

Get a code

How to plan for errors?: **Bug fix, re-distribute** (see Packaging)

[9] *Multiresolution gray-scale and rotation invariant texture classification with local binary patterns*, Ojala, 2002 (TPAMI)

Continuous Integration (CI)

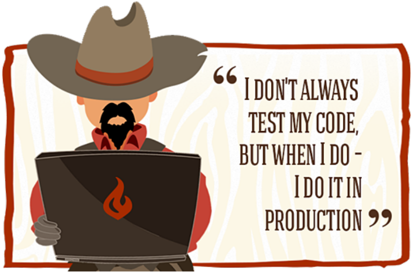

JUST IN CASE YOU'RE STILL NOT SURE WHETHER YOU'RE IN A SOFTWARE PROJECT
WAIT UNTIL YOU HEAR THIS:
geek & poke
ON MY MACHINE IT WORKS

What is Continuous Integration (CI)?

- CI helps you running your tests after pushing changes to your repository, *automatically*
- It is typically implemented using programmable *hooks* - every time you push code to your repository, the git server sends a message (triggers hook) to a service that runs a script on particular machine, on your behalf.
- We define and use this *script* to run our test units. Errors are reported back to the user (typically via an e-mail).
- A CI service can be used for more than just running (unit) tests. E.g.: they can be used to automatically calculate and publish software coverage, or update the package documentation.
- You can also use CI machinery to publish new versions of your project, but this is **not** continous integration *per se*, but rather continuous deployment (CD).

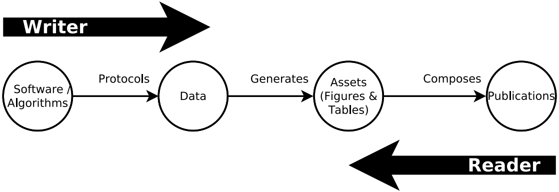
Testing and CI: Tooling

- If using Python, use **https://pytest.org** to define and run your test suite.
- Otherwise, just check Wikipedia!
- Continuous Integration:
  - At GitHub, there is a number of free-public-repositories services (e.g. Travis CI)
  - At GitLab, use GitLab CI (**https://docs.gitlab.com/ee/ci/**).
- To report test *coverage* results, try **https://coveralls.io**


“I DON'T ALWAYS TEST MY CODE, BUT WHEN I DO - I DO IT IN PRODUCTION”

Documentation

Documentation is there to help you remember, others to figure out.



- Data (what data is, how it was acquired, how to access it)
- Code (what your code does, how to install it)
  - Remember: Code is never self-documenting!
  - You can unit-test your documentation (where relevant)
- Instructions (how to take code and apply to the data)
  - Code usage examples may work as documentation
- Achievements (plots, tables, conclusions)

Document all you can!

Producing a README file for your project is great, but documentation may go beyond basic information. Here are some important areas often neglected:

1. Document command-line scripts with useful help messages, so you remember how to use them
2. Add documentation for user-side APIs
3. Comment your source code so you remember trick bits

Document all you can!

Producing a README file for your project is great, but documentation may go beyond basic information. Here are some important areas often neglected:

1. Document command-line scripts with useful help messages, so you remember how to use them
2. Add documentation for user-side APIs
3. Comment your source code so you remember trick bits

Tip

Choose a mark-up language and stick to it everywhere. If using Python, RestructuredText (`.rst` files) is probably the way to go. Use Sphinx **https://www.sphinx-doc.org** for the project documentation.

Note: Executable Reports (a.k.a. "notebooks")

Executable reports (ER) or papers are files that include embedded software that can be executed by the user to reproduce the presented results.

+ An ER can be executed directly, without cut-n-paste intervention, as it would be for code available in a Sphinx documentation, for example;
+ An ER is self-documented, so one may include all steps to reproduce results including instructions for any required software installation.
+ Excellent tool for prototyping and exploration and sharing insights.

- An ER requires a larger code base to start executing.
- An ER is not extensible *per se*.
- An ER is more difficult to lint and tends to be more chaotically organised than a proper software package.
- Programming tools such as code auto-completion may be limited.

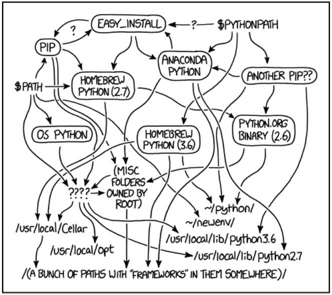
Executable Reports: Tooling

- Jupyter Notebook is a web application that combines integrated code and its output with visualization, mathematical equations, images, and text.
  - Supports dozens of programming languages
  - An Online Tutorial
- Kaggle is a platform that allows you to submit Jupyter Notebooks for hosted competitions, instead of software packages
- CodaLab is a platform allows one to reproducible experiments and create executable papers using the concept of worksheets.
- Binder is an open-source project allowing one to bind a JupyterNotebook on GitHub with a computing node to run experiments and generate results reproducibly.

Packaging and Deployment

Quickly deploying and switching environments is a time saver.

- Key points:
  - Tracking own build/run/test-time dependencies
  - Fast to create new environments for quick tests or deployments
- There are various standards for packaging code. It is very easy to get lost!



MY PYTHON ENVIRONMENT HAS BECOME SO DEGRADED THAT MY LAPTOP HAS BEEN DECLARED A SUPERFUND SITE.

Choosing How to Package

The idea of bundling software is not new.

- In the past, mostly used to ship (base) Operating Systems
- Next in the pipe was providing software/updates to users:
  - Linux/Debian (dpkg/apt), Linux/RedHat (rpm/yum), Free-BSD/port, Snap (**https://snapcraft.io**)
  - MacOS/MacPorts or Homebrew: **https://brew.sh**
  - Windows/Chocolatey: **https://chocolatey.org**
- As complementary part of a programming language itself:
  - Python: **https://packaging.python.org**
  - NPM/Javascript: **https://www.npmjs.com**
  - Ruby Gems: **https://rubygems.org**
- More recently, we saw the apperance of OS- and language-independent package managers, such as Conda **https://docs.conda.io/en/latest/**

Choosing How to Package

The idea of bundling software is not new.

- In the past, mostly used to ship (base) Operating Systems
- Next in the pipe was providing software/updates to users:
  - Linux/Debian (dpkg/apt), Linux/RedHat (rpm/yum), Free-BSD/port, Snap (**https://snapcraft.io**)
  - MacOS/MacPorts or Homebrew: **https://brew.sh**
  - Windows/Chocolatey: **https://chocolatey.org**
- As complementary part of a programming language itself:
  - Python: **https://packaging.python.org**
  - NPM/Javascript: **https://www.npmjs.com**
  - Ruby Gems: **https://rubygems.org**
- More recently, we saw the apperance of OS- and language-independent package managers, such as Conda **https://docs.conda.io/en/latest/**

Tip

If distributing to a wide audience, choose OS-agnostic package managers (`conda` seems popular these days). For Python-only projects, `pip` seems to be the *de facto* standard.

Other packaging technologies

- (Docker) Containers[10] have been lately promoted as good for reproducibility in AI:
  - You may have to re-imagined you project as a set of "micro"-services (training, inference, etc)
  - Works well, e.g., for shipping **trained** models
  - Tooling: **https://pachyderm.io**, **https://www.kubeflow.org**
- Virtual Machines (VM) go a step further than Docker, and ship a whole OS within an image (kernel + libc).
  - There is some heavy machinery involved in simulating hardware, with with modern CPU advances
  - Same issues, as with dockers, for setting up large scale orchestration, but with less tools.

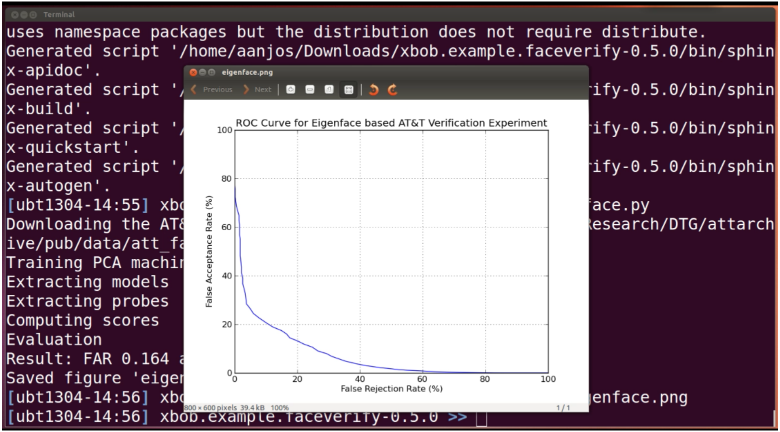
Tip: often, using these is not necessary

Docker and VMs will work best if implemented on the top of conventional packaging (i.e. create container from package).

[10] https://www.docker.com/resources/what-container

Example Deployment

Basic Face Recognition Example (old, but "gold")


Terminal
eigenface.png
Previous
Next
ROC Curve for Eigenface based AT&T Verification Experiment
False Acceptance Rate (%)
False Rejection Rate (%)
0
20
40
60
80
100
0
20
40
60
80
100
800 × 600 pixels 39.4 kB 100%
1 / 1

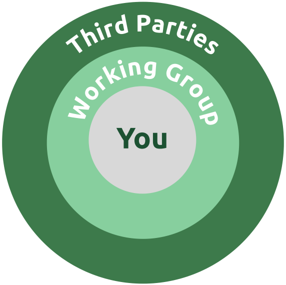
```
uses namespace packages but the distribution does not require distribute.
Generated script '/home/aanjos/Downloads/xbob.example.faceverify-0.5.0/bin/sphin
x-apidoc'.
Generated script '/                                    rify-0.5.0/bin/sphin
x-build'.
Generated script '/                                    rify-0.5.0/bin/sphin
x-quickstart'.
Generated script '/                                    rify-0.5.0/bin/sphin
x-autogen'.
[ubt1304-14:55] xbo                                    face.py
Downloading the AT&                                    Research/DTG/attarch
ive/pub/data/att_fa
Training PCA machin
Extracting models
Extracting probes
Computing scores
Evaluation
Result: FAR 0.164 a
Saved figure 'eigen
[ubt1304-14:56] xbo                                    genface.png
[ubt1304-14:56] xbob.example.faceverify-0.5.0 >>
```

Poll

On your AI projects, have you ever:

- □ Published open-access data?
- □ Used a tool for workflow/pipeline management such as Dask, Snakemake or other?
- □ Used Git or other version control system in your AI projects?
- □ Shared your AI project code through a social code sharing platform such as GitHub?
- □ Designed a test suite for your code?
- □ Documented your AI project beyond the "README" file, to add code comments, API documentation or baseline results?
- □ Bothered in packaging your AI project so it is easy to install?

You are in the center



- **You**, first!
  - Improved project structure and organization
  - Easy to replay analysis and generate results *after* changing mistakes
  - Easy to *extend* study to different tools and data
- **Collaborators**:
  - Closer interaction between collaborators
  - Scientific reports practically “write themselves"
  - Easy to pass-on work to colleagues
- **Others**:
  - Increased visibility (researchers)

Potential Impact

Boost your research **impact (visibility)**:

- **Lower entrance barrier** to your publications
- The current number of reproducible papers is **rather small** - you have a clear chance to stand out today:
  - Only **10% of TIP** papers provide source code[11].
- Statistically, your work is **more valuable** if it is RR:
  - **13 out of the top 15 most cited** articles in TPAMI or TIP provide (at least) source code
  - The average number of citations for papers that provide source-code in TIP is **7 fold** that of papers that do not.

[5] *Code Sharing is Associated with Research Impact in Image Processing*, Patrick Vandewalle, 2012

At Idiap:

We handle Continual/Full Reproducibility (free software, easy deployment) in 3 ways:

- Develop and maintain a basic set of tools that we all share: database protocol APIs, signal (image, audio) processing, machine learning, evaluation - **https://www.idiap.ch/software/bob**
- Provide most of our **databases publicly**, free of charge
- Provide end-user **applications** wrapped in an **easy to deploy** packaging system that users (potential citers) can download, install and extend

Takeaway Message

Organize yourself so you are ***always*** *doing Reproducible Research*

Benefits:

- Work is always kept reproducible
- Easy to replay analysis in case of errors
- Easy to change code to run different analysis, ML techniques or on different data
- If working collaboratively:
  - Team members help each other in case of problems
  - New colleagues can start (nearly) immediately to produce high-quality results.

Practical Guide to Better Reproducibility

. . . or how far should I go?

1. Make sure (most of) data you use is publicly available
2. Start using a Version Control system for your projects (e.g. Git):
   - 2.1 Pick a Git server and maintain all of your code there (e.g. GitHub or GitLab)
   - 2.2 Commit soon and often so others can follow
3. Implement basic documentation that explains how to use your software (README)
4. Make it a habit to re-use common code through your projects
5. Implement software packaging:
   - 5.1 Data reading/loading iterators, respecting specific (repeatable) evaluation protocols
   - 5.2 Analysis tools that produce comparable results
   - 5.3 Finally, package your code as well (published article). Tip: Adding trained models may help future users save time.
6. Be methodic: report results (at least) using pre-defined evaluation protocols and metrics from base packages
7. Provide basic documentation with your package showing how to reproduce your (article) results
8. If you want to maintain your package for a longer period of time, include a test suite and improve code coverage
9. Expand documentation to the CLI and the API
10. Setup a CI system (e.g. Travis CI)
11. Working example: **https://github.com/idiap/mai-m05-ex6**

Thank you for your attention!

Dr. André Anjos (andreanjos.org)

May 3, 2021