A Practical Guide to Reproducible ML Research

ICAI Summer School - 2025

Philipp Hager, July 1st 2025

Waterval - M.C. Escher, 1961

About me

I'm a 4th year PhD student with Maarten de Rijke and Onno Zoeter at the IRLab (UvA) and the ICAI Mercury Machine Learning Lab (Booking.com).

Previously

- Recommender systems at Blinkist, Berlin
- M.Sc. Hasso-Plattner Institute, Potsdam
- B.Sc. University of Applied Sciences, Düsseldorf

Research interests

Information retrieval, unbiased learning-to-rank, and user simulation

Acknowledgments

Towards reproducible machine learning research in natural language processing
SIGIR 2022 tutorial by Ana Lucic, Maurits Bleeker, Maarten de Rijke, Koustuv Sinha, Sami Jullien, Robert Stojnic

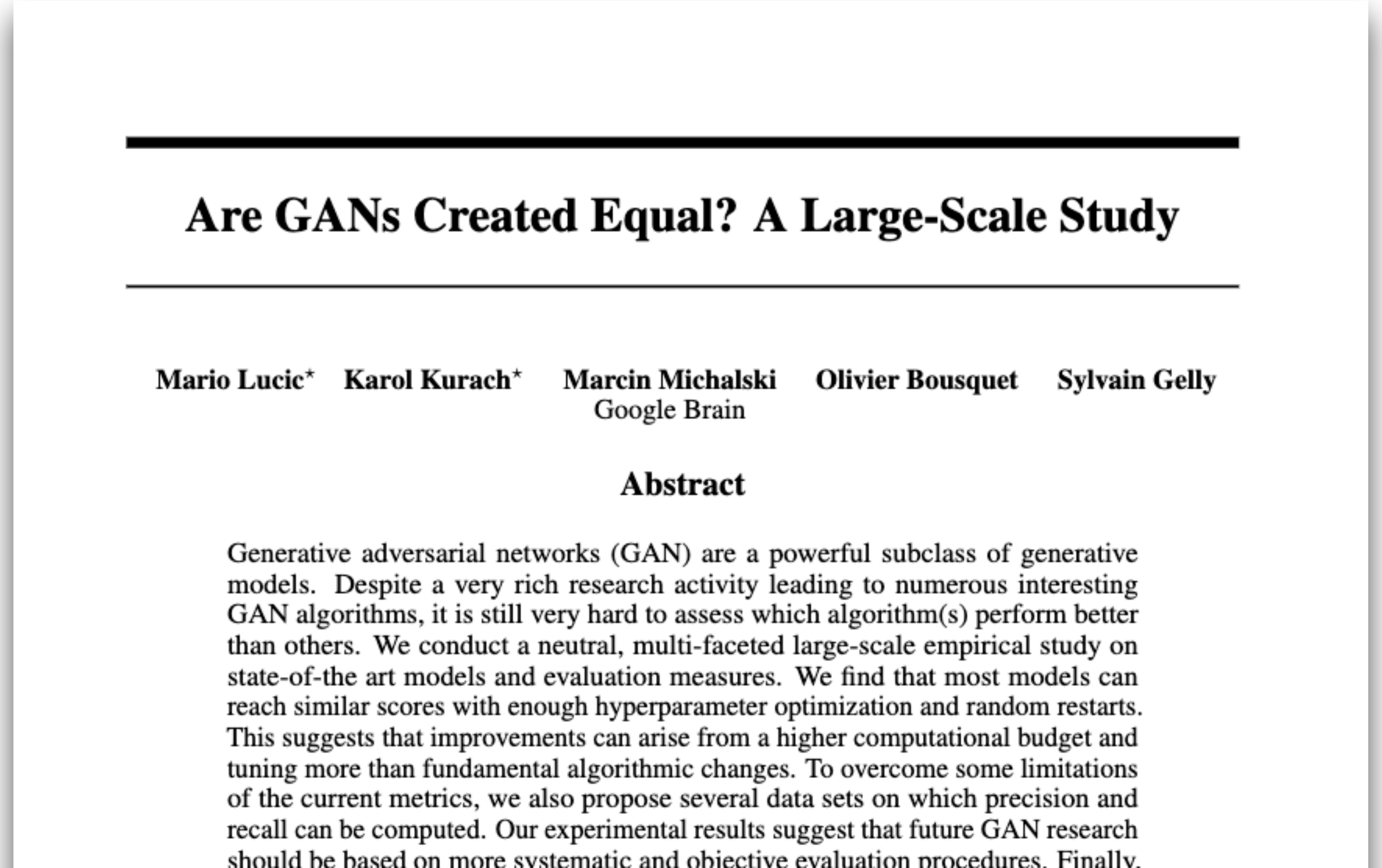
Why is reproducibility important?

Crisis? What crisis?

Generative Adversarial Networks (2018)

A comparison of seven popular GAN methods [1] found that *"most models can reach similar scores with enough **hyperparameter optimization** and **random restarts**".*

The authors found **no GAN extension consistently outperformed** the original when controlling for compute budgets [2].



Are GANs Created Equal? A Large-Scale Study

Mario Lucic* Karol Kurach* Marcin Michalski Olivier Bousquet Sylvain Gelly
Google Brain

Abstract

Generative adversarial networks (GAN) are a powerful subclass of generative models. Despite a very rich research activity leading to numerous interesting GAN algorithms, it is still very hard to assess which algorithm(s) perform better than others. We conduct a neutral, multi-faceted large-scale empirical study on state-of-the art models and evaluation measures. We find that most models can reach similar scores with enough hyperparameter optimization and random restarts. This suggests that improvements can arise from a higher computational budget and tuning more than fundamental algorithmic changes. To overcome some limitations of the current metrics, we also propose several data sets on which precision and recall can be computed. Our experimental results suggest that future GAN research should be based on more systematic and objective evaluation procedures. Finally,

[1] Lucic, Mario, et al. "Are gans created equal? a large-scale study." In NeurIPS 2018.
[2] Goodfellow, Ian J., et al. "Generative adversarial nets." In NeurIPS 2014.

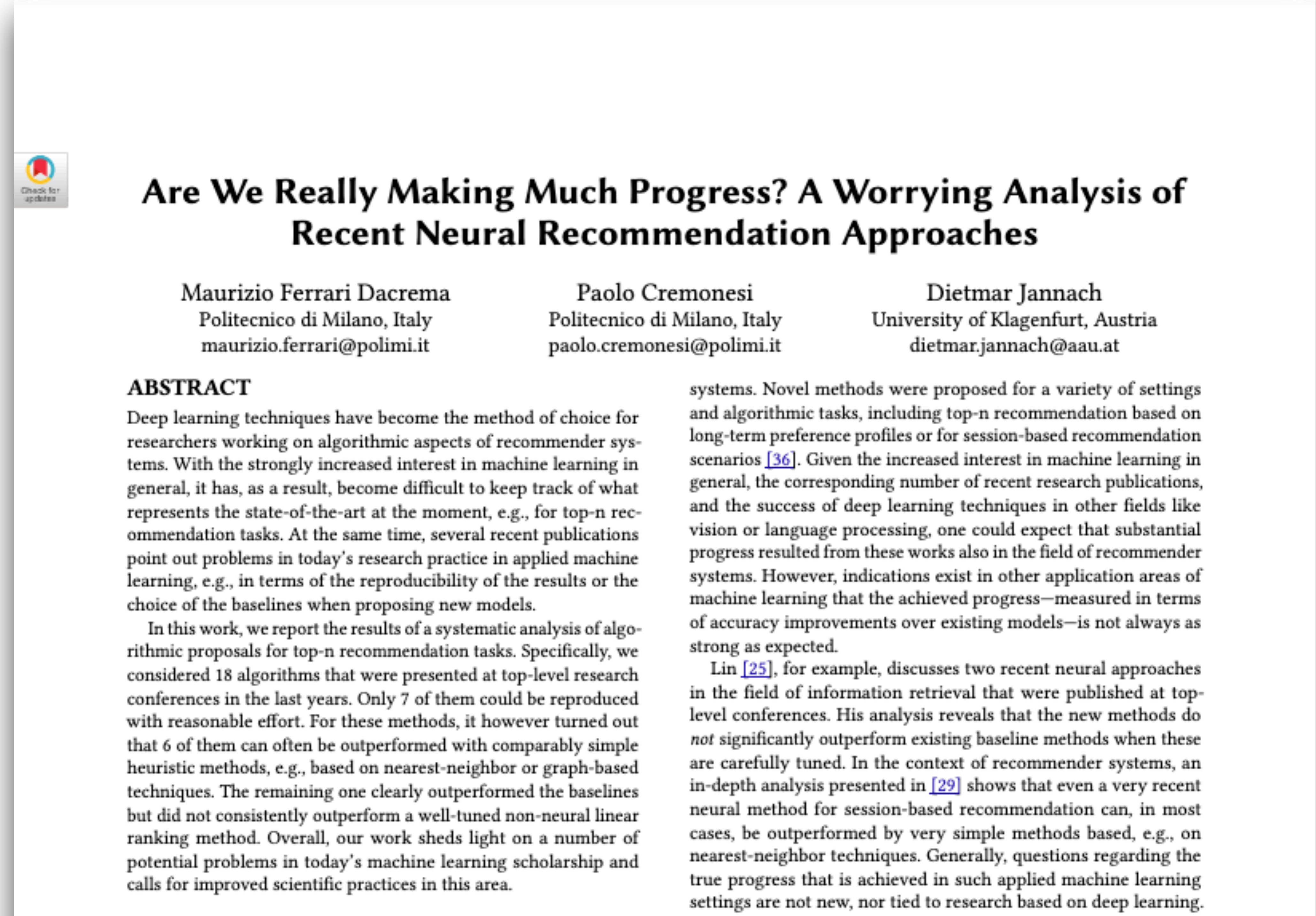
Neural Recommender Systems (2019)

Survey of 18 neural top-n recommender systems published at RecSys, KDD, SIGIR, TheWebConf between 2015 and 2018.

Only 7/18 papers could be **reproduced**.

6/7 papers were outperformed by simple **heuristics** (KNN and graph-based methods).

One paper outperformed heuristics but not consistently a well-tuned linear model.



Are We Really Making Much Progress? A Worrying Analysis of Recent Neural Recommendation Approaches

Maurizio Ferrari Dacrema
Politecnico di Milano, Italy
maurizio.ferrari@polimi.it

Paolo Cremonesi
Politecnico di Milano, Italy
paolo.cremonesi@polimi.it

Dietmar Jannach
University of Klagenfurt, Austria
dietmar.jannach@aau.at

ABSTRACT

Deep learning techniques have become the method of choice for researchers working on algorithmic aspects of recommender systems. With the strongly increased interest in machine learning in general, it has, as a result, become difficult to keep track of what represents the state-of-the-art at the moment, e.g., for top-n recommendation tasks. At the same time, several recent publications point out problems in today's research practice in applied machine learning, e.g., in terms of the reproducibility of the results or the choice of the baselines when proposing new models.

In this work, we report the results of a systematic analysis of algorithmic proposals for top-n recommendation tasks. Specifically, we considered 18 algorithms that were presented at top-level research conferences in the last years. Only 7 of them could be reproduced with reasonable effort. For these methods, it however turned out that 6 of them can often be outperformed with comparably simple heuristic methods, e.g., based on nearest-neighbor or graph-based techniques. The remaining one clearly outperformed the baselines but did not consistently outperform a well-tuned non-neural linear ranking method. Overall, our work sheds light on a number of potential problems in today's machine learning scholarship and calls for improved scientific practices in this area.

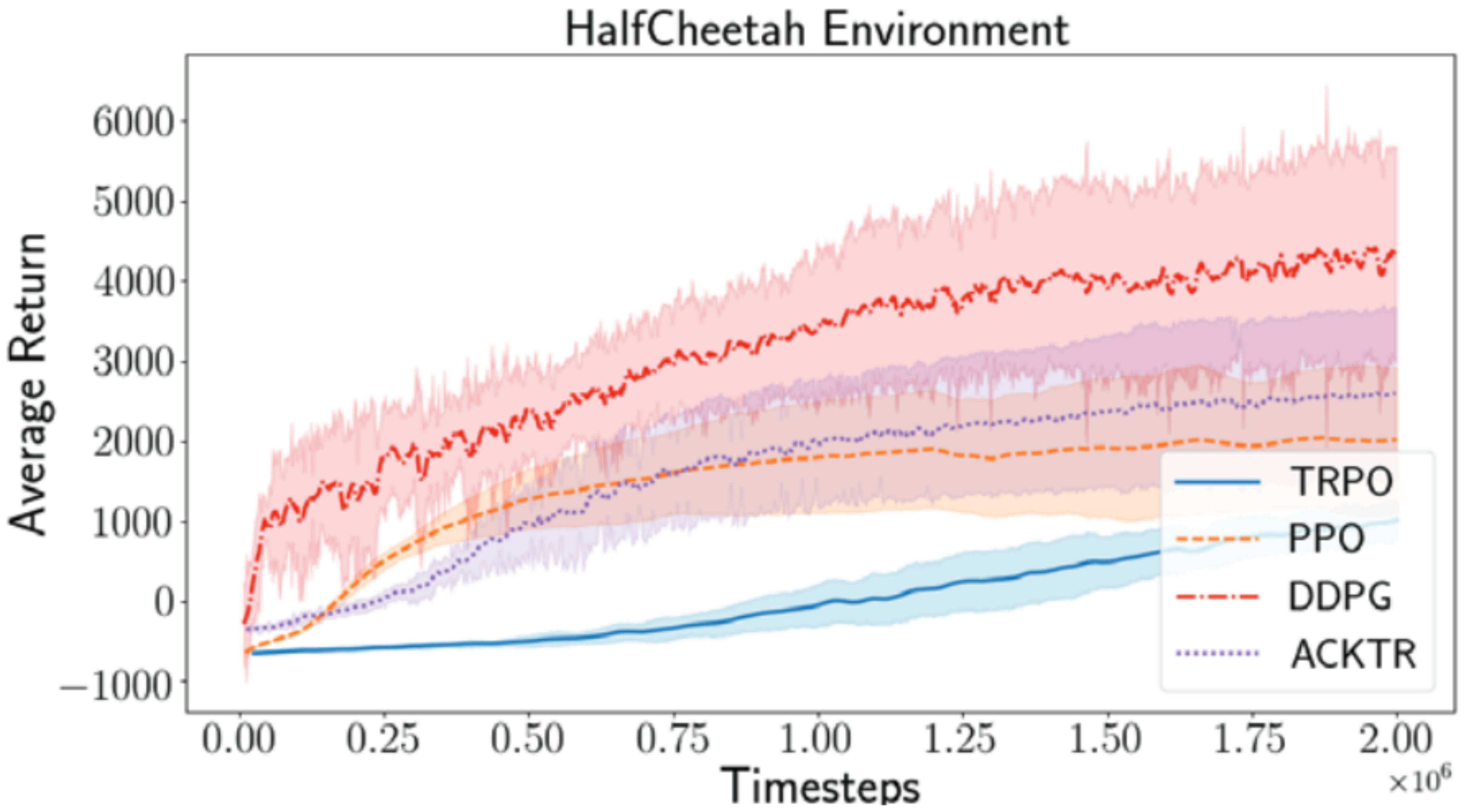
systems. Novel methods were proposed for a variety of settings and algorithmic tasks, including top-n recommendation based on long-term preference profiles or for session-based recommendation scenarios [36]. Given the increased interest in machine learning in general, the corresponding number of recent research publications, and the success of deep learning techniques in other fields like vision or language processing, one could expect that substantial progress resulted from these works also in the field of recommender systems. However, indications exist in other application areas of machine learning that the achieved progress—measured in terms of accuracy improvements over existing models—is not always as strong as expected.

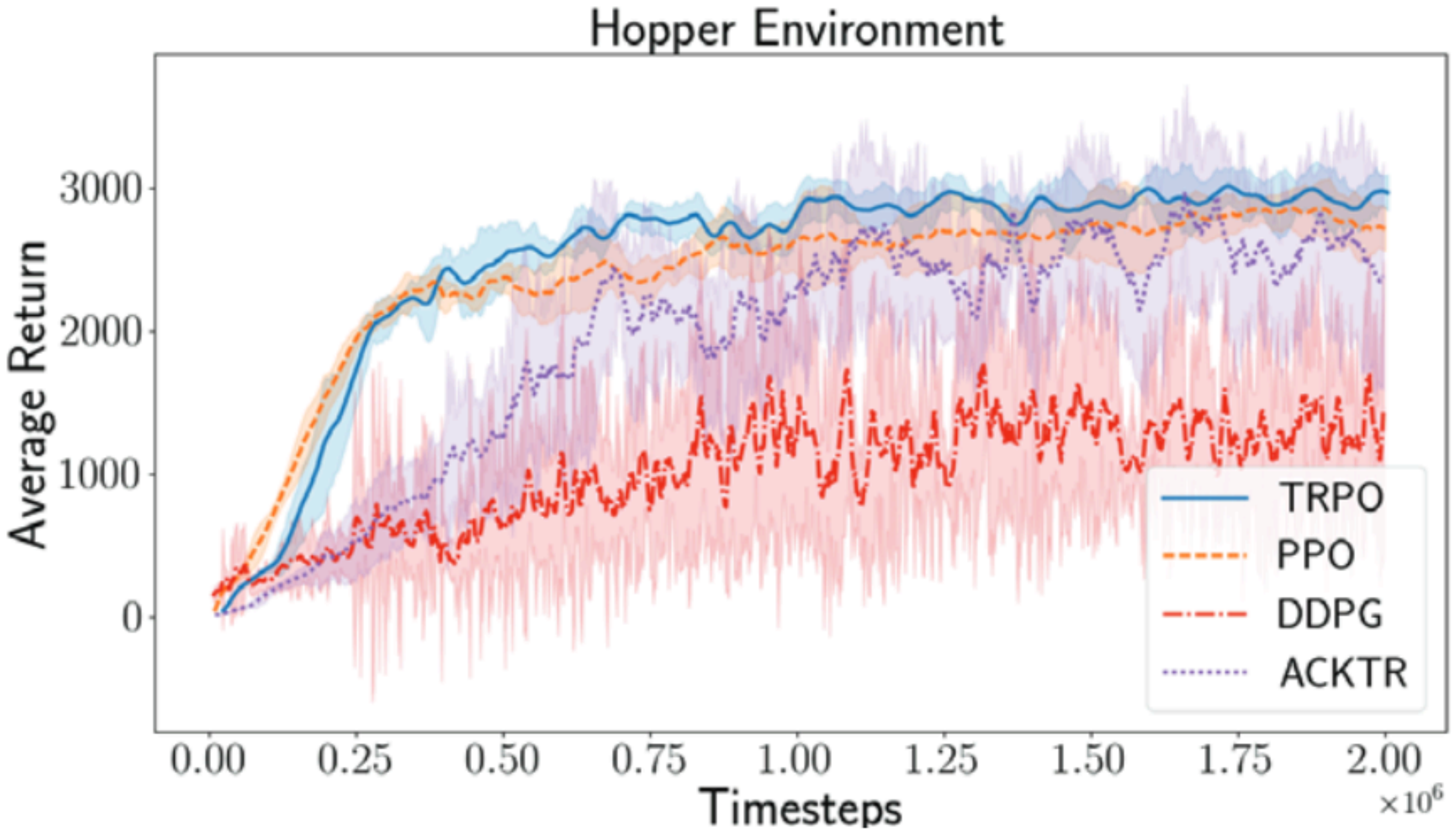
Lin [25], for example, discusses two recent neural approaches in the field of information retrieval that were published at top-level conferences. His analysis reveals that the new methods do *not* significantly outperform existing baseline methods when these are carefully tuned. In the context of recommender systems, an in-depth analysis presented in [29] shows that even a very recent neural method for session-based recommendation can, in most cases, be outperformed by very simple methods based, e.g., on nearest-neighbor techniques. Generally, questions regarding the true progress that is achieved in such applied machine learning settings are not new, nor tied to research based on deep learning.

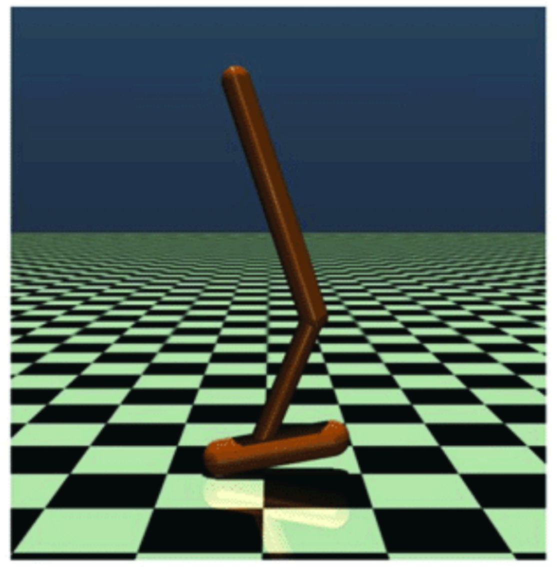
[1] Ferrari Dacrema, Maurizio, Paolo Cremonesi, and Dietmar Jannach. Are we really making much progress? A worrying analysis of recent neural recommendation approaches. In RecSys 2019.

Deep Reinforcement Learning (2018)

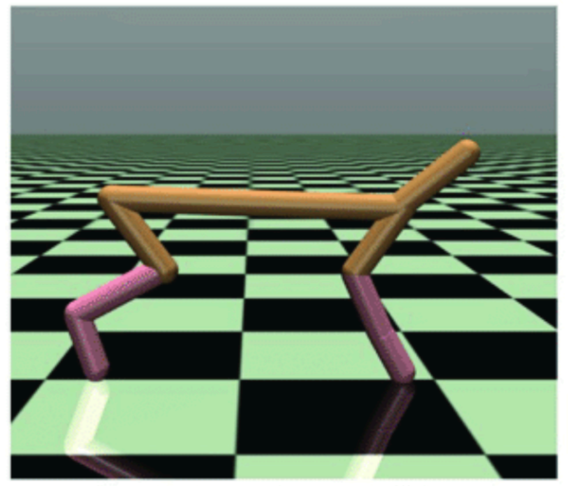
Henderson et al. [1] highlight reproducibility challenges for policy gradient methods:







Hopper



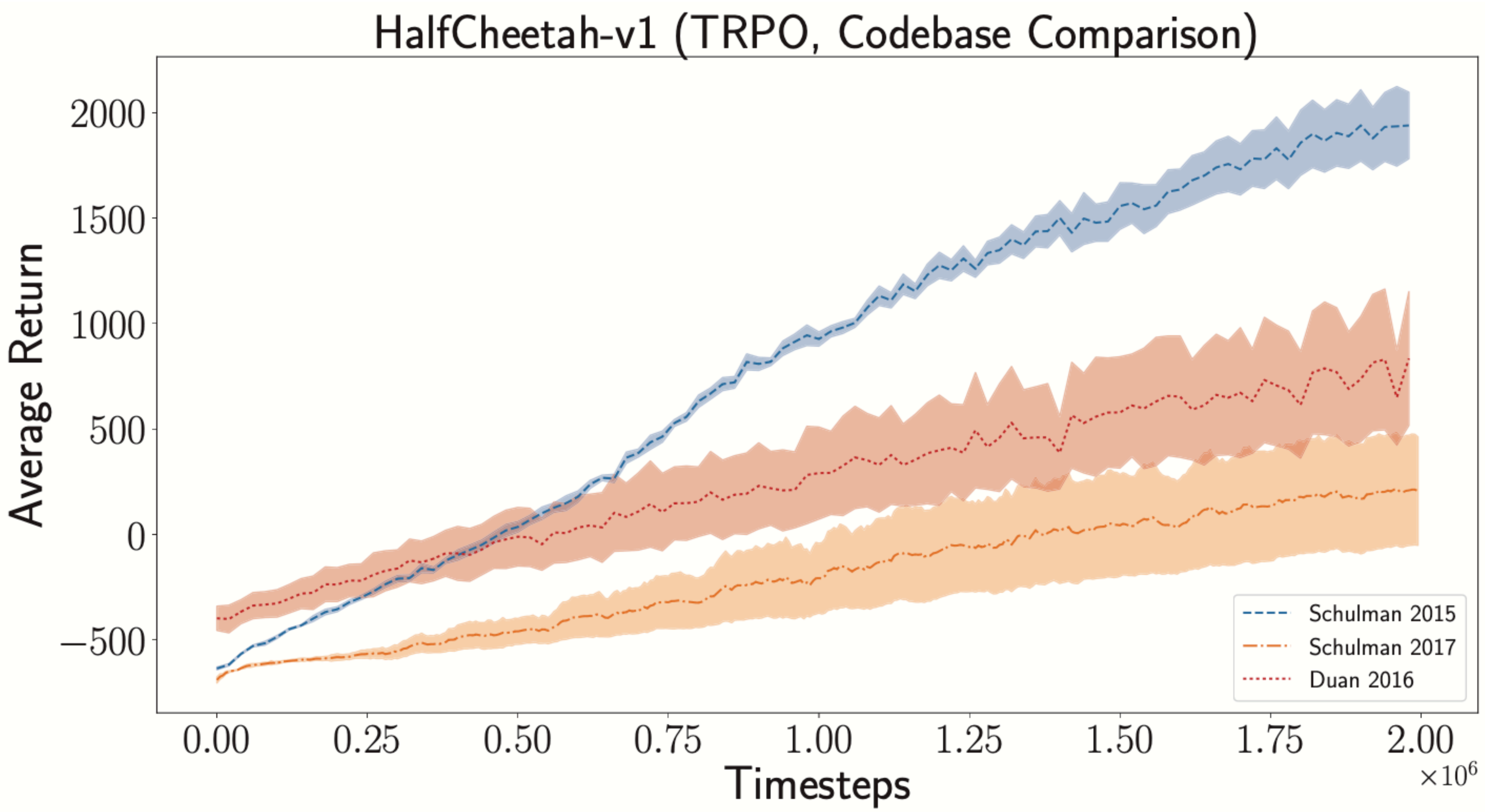
Half-Cheetah

Large performance differences between baselines on related MuJoCo simulations [Figure 4, 1]

[1] Henderson, Peter, et al. Deep reinforcement learning that matters. In AAAI 2018.

Deep Reinforcement Learning (2018)

Henderson et al. [1] highlight reproducibility challenges for policy gradient methods:

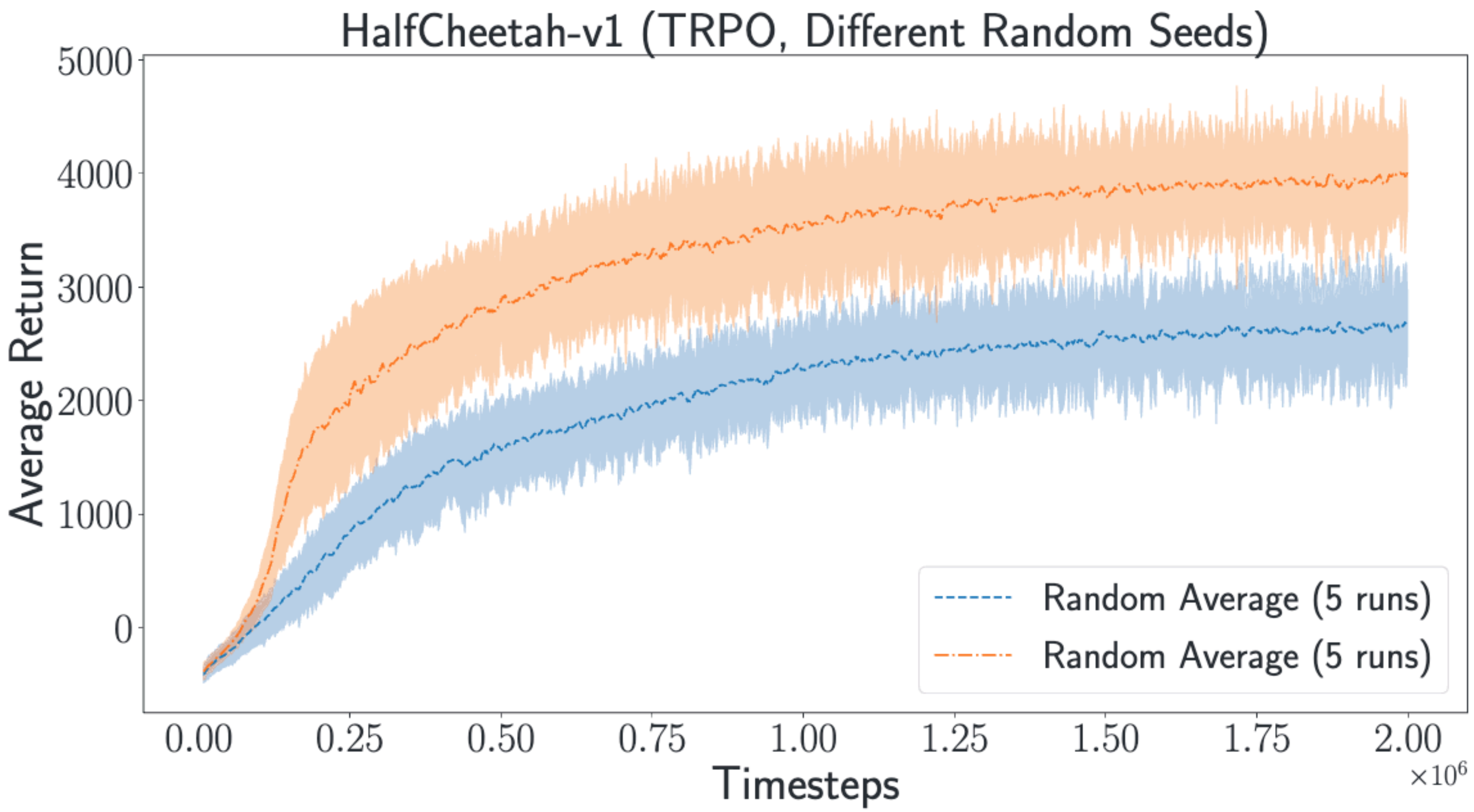


Large performance differences between implementations of the same method due to modeling decisions such as activations, model architecture, ... [Figure 6, 1]

[1] Henderson, Peter, et al. Deep reinforcement learning that matters. In AAAI 2018.

Deep Reinforcement Learning (2018)

Henderson et al. [1] highlight reproducibility challenges for policy gradient methods:



Runs of the exact same hyperparameters over different random seeds can appear like different distributions [Figure 5, 1].

[1] Henderson, Peter, et al. Deep reinforcement learning that matters. In AAAI 2018.

And many more examples…

Reproducibility has been a problem in, e.g.:

- Metric learning [1]
- Deep Bandits [2]
- Computer vision [3]
- Forecasting [4]
- Natural language processing [5]
- Information retrieval [6]

[1] Musgrave, Kevin, Serge Belongie, and Ser-Nam Lim. A metric learning reality check. In ECCV 2020.
[2] Riquelme, Carlos, George Tucker, and Jasper Snoek. Deep bayesian bandits showdown: An empirical comparison of bayesian deep networks for thompson sampling. In ICLR 2018.
[3] Bouthillier, Xavier, César Laurent, and Pascal Vincent. Unreproducible research is reproducible. In ICML 2019.
[4] Makridakis, Spyros, Evangelos Spiliotis, and Vassilios Assimakopoulos. Statistical and Machine Learning forecasting methods: Concerns and ways forward. In PloS 2018.
[5] Belz, Anya, et al. A systematic review of reproducibility research in natural language processing. In EACL 2021.
[6] Lin, Jimmy. The neural hype and comparisons against weak baselines. In SIGIR Forum 2019.

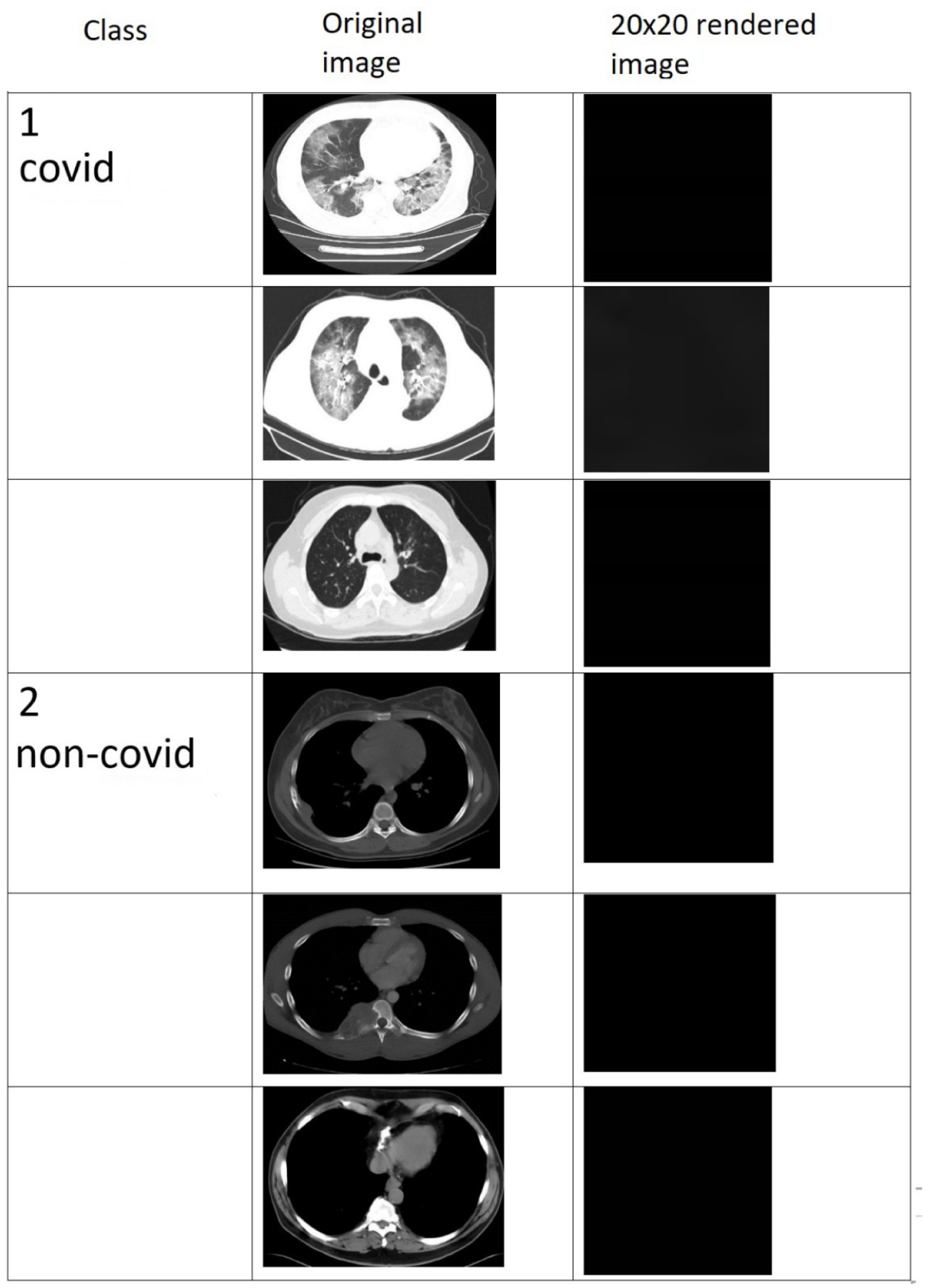
...also outside of computer science

ML is being applied in medicine, chemistry, biology...

During COVID multiple papers proposed COVID **classifiers based on chest X-Rays [1].**

A follow-up study found COVID could be detected **above chance (67%) just from the background [2].**

In many cases, this is a problem of **data leakage**, e.g., similar patients or similar instruments in train/test [3,4].



Chest X-Rays and backgrounds from [2]

[1] Khan, Asif Iqbal et al. "CoroNet: A deep neural network for detection and diagnosis of COVID-19 from chest x-ray images." Computer methods and programs in biomedicine 196 (2020).
[2] Dhar, Sanchari, and Lior Shamir. "Evaluation of the benchmark datasets for testing the efficacy of deep convolutional neural networks." Visual informatics 5.3 (2021): 92-101.
[3] Kapoor, Sayash, and Arvind Narayanan. "Leakage and the reproducibility crisis in machine-learning-based science." Patterns 4.9 (2023).
[4] Ball, Philip. Is AI leading to a reproducibility crisis in science? Nature, 2023: https://www.nature.com/articles/d41586-023-03817-6

Why care about reproducibility?

"It is a truism within the community that at least one clear win is needed for acceptance at a top venue.

Yet, a moment of reflection recalls that ***the goal of science is not wins, but knowledge*** *[1]."*

[1] Sculley, David, et al. Winner's curse? On pace, progress, and empirical rigor. In ICLR workshop 2018.

What is reproducibility?

A lot of fuzzy terminology

ACM Definitions

Repeatability
Same team, same experimental setup

Reproducibility
Different team, same experimental setup

Replicability
Different team, different experimental setup

[1] Artifact Review and Badging Version 1.1 - August 24, 2020: https://www.acm.org/publications/policies/artifact-review-and-badging-current

Other conferences, other definitions ...

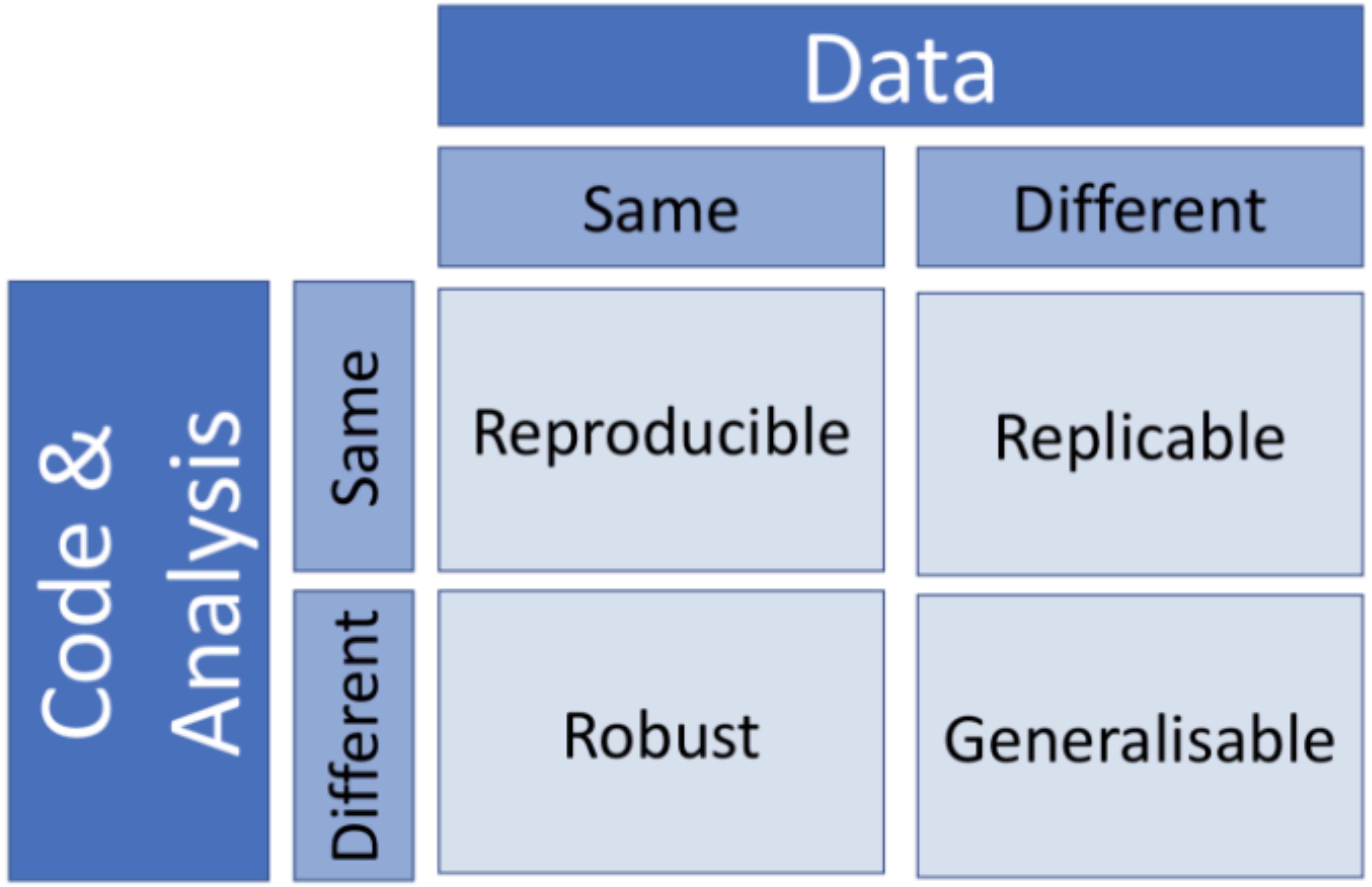
NeurIPS definitions

Reproducible
Same experimental setup, same data

Replicable
Same experimental setup, different data

Robust
Different experimental setup, same data

Generalizable
Different experimental setup, different data



Defining reproducibility at NeurIPS [1]

[1] Pineau, Joelle, et al. Improving reproducibility in machine learning research. In JMLR 2021.

… but similar notions

As Gundersen [1] observes: *“reproducibility is an elusive concept”, but some ideas are similar:*

Re-run code
The published code/setup is executable and gives similar results

Re-implement idea
A method can be implemented and gives similar results

Idea/lesson generalizes (actual progress)
We can draw similar conclusions in new experimental setups

[1] Gundersen, Odd Erik. The fundamental principles of reproducibility. In Philosophical Transactions of the Royal Society 2021.

What makes papers irreproducible?

I. Documentation & Communication

Insufficient documentation

Gundersen and Kjensmo [1] surveyed 400 papers (2013 - 2016) and found that **documentation practices in AI render most reported research results irreproducible***, e.g.:

Method
Formulate problem statements (47%), objective (22%), or research questions (6%)

Results
Release train set (56%), test set (30%), or results (4%)

Experiments
Describe the setup (69%), hardware specs (27%), or release code (8%)

[1] Gundersen, Odd Erik, and Sigbjørn Kjensmo. State of the art: Reproducibility in artificial intelligence. In AAAI 2018.
* Disclaimer: The authors searched for explicit terms. Thus, the numbers are probably too low.

Insufficient communication

Raff [1] implemented 255 papers (1984 - 2017) from documentation alone (no published code used).

Overall, 63.5% papers could be reproduced*.

50/255 authors were contacted, of which 52% replied:

- 22/26 papers of authors who replied could be reproduced (85%).
- Only 1/24 paper of authors who did not reply could be reproduced (4%).

Current **publishing structures do not incentivize follow-up support** once a paper is published.

Table 1: Significance test of which paper properties impact reproducibility. Results significant at $\alpha \leq 0.05$ marked with"*".

Feature	p-value
Year Published	0.964
Year First Attempted	0.674
Venue Type	0.631
Rigor vs Empirical*	1.55×10^{-9}
Has Appendix	0.330
Looks Intimidating	0.829
Readability*	9.68×10^{-25}
Algorithm Difficulty*	2.94×10^{-5}
Pseudo Code*	2.31×10^{-4}
Primary Topic*	7.039×10^{-4}
Exemplar Problem	0.720
Compute Specified	0.257
Hyperparameters Specified*	8.45×10^{-6}
Compute Needed*	8.75×10^{-5}
Authors Reply*	6.01×10^{-8}
Code Available	0.213
Pages	0.364
Publication Venue	0.342
Number of References	0.740
Number Equations*	0.004
Number Proofs	0.130
Number Tables*	0.010
Number Graphs/Plots	0.139
Number Other Figures	0.217
Conceptualization Figures	0.365
Number of Authors	0.497

[1] Raff, Edward. "A step toward quantifying independently reproducible machine learning research." In NeurIPS 2019.
* A paper was reproducible if 75% of its claims could be verified, see [1, Section 2] for details.

II. Scientific method

Hypothesis testing

Only 47% of AI papers included a problem statement [1]. But let's be honest, we also often **start projects with coding / experiments right away.**

That, however, can lead to:

- **Unclear research questions (RQs)**
- **Wrong conclusions**
- **Wasted time, effort, and computational power**

Formulate (at least an initial version) the RQs before starting experiments.

[1] Gundersen, Odd Erik, and Sigbjørn Kjensmo. State of the art: Reproducibility in artificial intelligence. In AAAI 2018.

"Grad Student Descent"

1. Begin with a baseline

2. Try random modifications, e.g., model architecture and hyperparams

3. Iterate until local optima / promising results

4. Post-hoc rationalize why method works

Ablation studies help, but do not avoid the core issue of overfitting the research process.

[1] Gencoglu, Oguzhan, et al. "HARK Side of Deep Learning–From Grad Student Descent to Automated Machine Learning." arXiv preprint arXiv:1904.07633 (2019).

Problematic hypothesis testing

Cherry-picking: Only report results that support your hypothesis.

P-Hacking: Analyze the results in different ways (e.g., including/excluding covariates) until you find a significant result.

Fishing expeditions: Indiscriminately examine associations between variables without intending to test a priori hypothesis.

Hypothesizing After the Results are Known (HARKing): Find a significant result and construct your hypothesis retroactively. Note that this is not the same as an exploratory analysis.

[1] Andrade, Chittaranjan. HARKing, cherry-picking, p-hacking, fishing expeditions, and data dredging and mining as questionable research practices. The Journal of clinical psychiatry 2021.

Statistical testing

Comparing the means of two models is not enough to conclude model A is better than model B. Especially in ML, we often **obtain significant differences** by chance [1, 2].

Here are a few things to keep in mind [3, 4]:

- Compare model runs across seeds and datasets
- Formulate a null hypothesis per dataset
- Correct for multiple comparisons when comparing multiple models
- Perform a power analysis to check if you need more seeds [5]

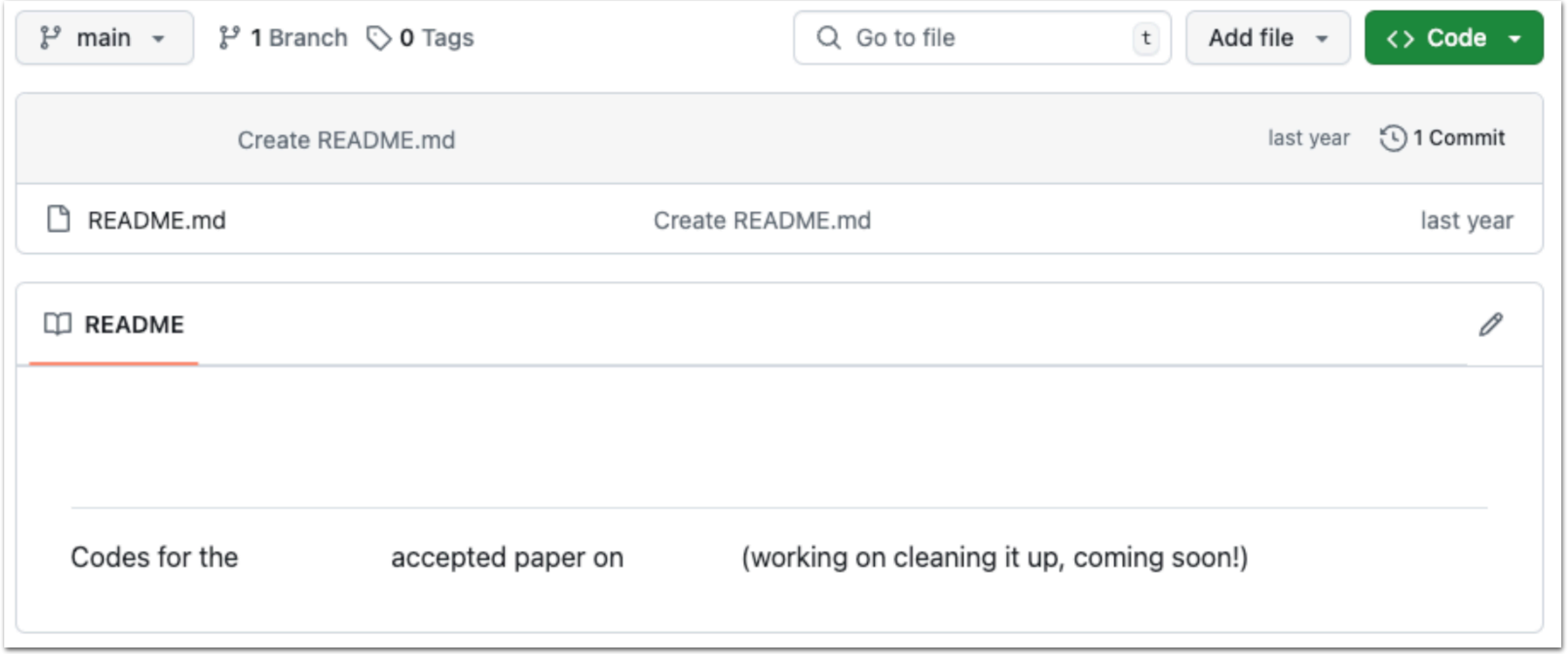
Report the used tests, the significance level, and add confidence intervals

[1] Reimers, Nils, and Iryna Gurevych. Why comparing single performance scores does not allow to draw conclusions about machine learning approaches. arXiv preprint arXiv:1803.09578 (2018).
[2] Dehghani, Mostafa, et al. "The benchmark lottery." arXiv preprint arXiv:2107.07002 (2021).
[3] Urbano, Julián, Harlley Lima, and Alan Hanjalic. Statistical significance testing in information retrieval: an empirical analysis of type I, type II and type III errors. In SIGIR 2019.
[4] Smucker, Mark D., James Allan, and Ben Carterette. A comparison of statistical significance tests for information retrieval evaluation. In CIKM 2007.
[5] Colas, Cédric, Olivier Sigaud, and Pierre-Yves Oudeyer. "How many random seeds? statistical power analysis in deep reinforcement learning experiments."

III. Code & data

How NOT to publish code

Implementation details. The source code to reproduce the findings from the paper is available at: https://github.com/



Not publishing all necessary code & data

In [3], **12/21 papers linked a repository**. **In 2/12 cases**, that repository was **empty or non-existent.** Even if code is published, it is **often incomplete** [1, 2, 3]:

- **Datasets:** Including splits and preprocessing steps
- **Baselines:** Including code and hyperparameter tuning
- **Method:** All details, final hyperparameters, and random seeds
- **Evaluation protocol & visualizations**
- **Dependencies:** List of all dependencies with exact versions
- **Scripts:** All scripts used to orchestrate the project
- **Stale URLs:** Links for code and data stop working…

See the **NeurIPS guidelines for publishing research code!**

[1] Ferrari Dacrema, Maurizio, Paolo Cremonesi, and Dietmar Jannach. Are we really making much progress? A worrying analysis of recent neural recommendation approaches. In RecSys 2019.
[2] Ferrari Dacrema, Maurizio, et al. A troubling analysis of reproducibility and progress in recommender systems research. In TOIS 2021.
[3] Shehzad, Faisal, and Dietmar Jannach. Everyone'sa winner! on hyperparameter tuning of recommendation models. In RecSys 2023.

Not polishing code

In most cases, papers defer to the code for exact details. However, **code quality impacts understanding and, thus, reproducibility**. Code readability can be impacted by, a.o:

- Inconsistent formatting
- Large amounts of commented code (e.g., commenting out different run options)
- Very long methods and complex file structures
- A high amount of redundancy
- Missing comments for unintuitive or difficult code
- Being too modular (not everything has to be a library)

…

How to polish code

An incomplete list of things, I think, makes code easier to understand:

- Follow the Python style-guide for naming stuff
- Use (strict) formatters: ruff, black, autopep8
- Remove unused code: isort, autoflake
- Catch bugs early with linters: flake8, pylama
- Remove commented code, look up old code in git
- Use environment managers with reproducible environments: uv, mamba, poetry
- Don't write files to disk unless absolutely necessary
- Write scripts that orchestrate your entire experiment

…

IV. Uncontrolled randomness

Randomness through design decisions

Pham et al. [2] found up to 10.8% accuracy differences between image classifier runs due to **algorithmic factors** that introduce stochasticity [1, 2]:

- Random weight initialization
- Stochastic operations (dropout, noisy activations)
- Data splitting, shuffling, batch ordering
- Random feature selection (e.g., in random forest)
- Hyperparameter tuning procedure (e.g., Bayesian methods)
- Sampled evaluation metrics [3]

Fix and report random seeds [4], release code, and datasets!

[1] Gundersen, Odd Erik, et al. Sources of irreproducibility in machine learning: A review. In arXiv:2204.07610 2022.
[2] Pham, Hung Viet, et al. "Problems and opportunities in training deep learning software systems: An analysis of variance." In ASE 2020.
[3] Krichene, Walid, and Steffen Rendle. "On sampled metrics for item recommendation." In KDD 2020.
[4] E.g., see: https://pytorch.org/docs/stable/notes/randomness.html

Algorithmic randomness in PyTorch

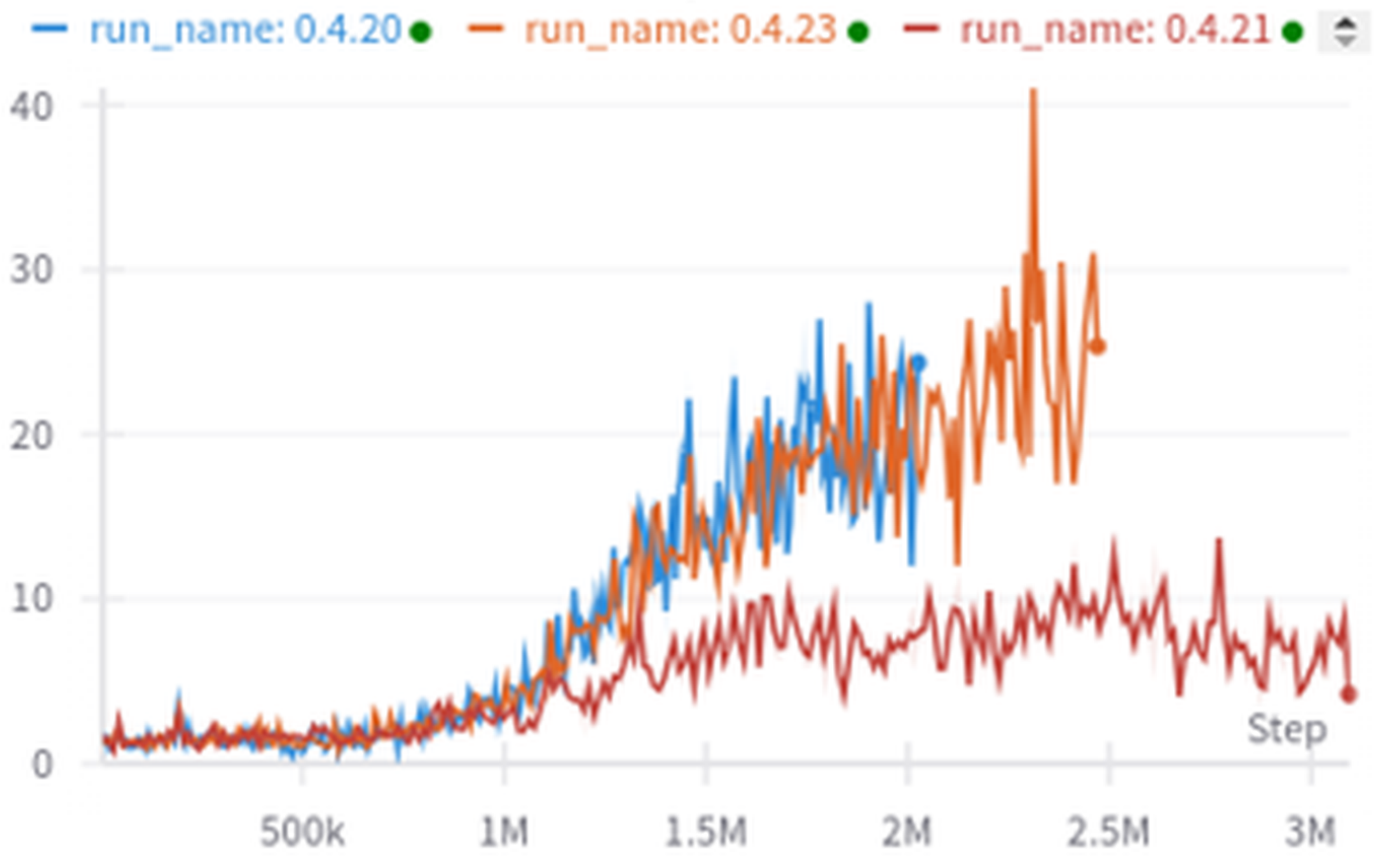
```python
def seed_everything(seed: int, workers: bool = False):
    # Python's built-in RNG: random.random(), random.choice(), etc.
    random.seed(seed)

    # NumPy's RNG:
    np.random.seed(seed)

    # PyTorch seed for CPU and CUDA [2]: weight init, dropout, sampling
    torch.manual_seed(seed)

    # PyTorch Lightning seed for new subprocesses (e.g., in DDP training)
    os.environ["PL_GLOBAL_SEED"] = str(seed)
    # PyTorch Lightning seeds for new DataLoader workers, otherwise [3]
    os.environ["PL_SEED_WORKERS"] = f"{int(workers)}"
```

[1] PyTorch Lightning, seed_everything: https://pytorch-lightning.readthedocs.io/en/1.7.7/_modules/pytorch_lightning/utilities/seed.html#pl_worker_init_function
[2] PyTorch Reproducibility: https://docs.pytorch.org/docs/stable/notes/randomness.html
[3] PyTorch seeding DataLoaders: https://docs.pytorch.org/docs/stable/notes/randomness.html#dataloader
[4] PyTorch deterministic algorithms: https://docs.pytorch.org/docs/stable/generated/torch.use_deterministic_algorithms.html#torch.use_deterministic_algorithms

Randomness through implementation

Pham et al. [2] found 2.9% accuracy differences after fixing seeds due to stochastic **implementation details** [1]:

- **Frameworks & versions** (Jax, PyTorch, TensorFlow)
- **Auto-selection of operations** (cuDNN & CUDA)
- **Parallel processing & random memory access**
- **Low-precision, quantization, and scheduling** [4]: $(A+B)+C \neq A+(B+C)$. Important when using, e.g., FlashAttention [5]



Reward of the same RL model across three different Jax versions [3]

Report software versions, hardware, and any computation optimizations.

[1] Gundersen, Odd Erik, et al. Sources of irreproducibility in machine learning: A review. In arXiv:2204.07610 2022.
[2] Pham, Hung Viet, et al. "Problems and opportunities in training deep learning software systems: An analysis of variance." In ASE 2020.
[3] Observation by Sami Jullien and Romain Deffayet
[4] Floating-point operations are not associative due to rounding: Goldberg, David. "What every computer scientist should know about floating-point arithmetic." ACM computing surveys (CSUR) 23.1 (1991): 5-48.
[5] Shah, Jay et al. FlashAttention-3: https://pytorch.org/blog/flashattention-3/

Disabling hardware optimization in PyTorch

Balance potential speed-ups and exactly replicable results:

```
def seed_everything(seed: int, workers: bool = False):
    …
    # cuDNN benchmarking CUDA convolution operations (slows down code)
    torch.backends.cudnn.benchmark = False

    # Use deterministic algorithms if possible (slows down code) [1]
    torch.use_deterministic_algorithms(True)
```

Not always advisable!

[1] PyTorch deterministic algorithms: https://docs.pytorch.org/docs/stable/generated/torch.use_deterministic_algorithms.html#torch.use_deterministic_algorithms

V. Baselines

Probably the number #1 complaint across reproducibility studies

Baselines

Unavailable baselines [1, 2]
E.g., copying results, parameters, or not including baseline code

Untuned baselines [1, 2, 4]
E.g., *we use the same parameters as X...*

Lack of simple baselines [1, 2, 3]
E.g., not comparing against sensible heuristics

Lack of strong baselines [1, 2, 4, 5, 6]
E.g., not comparing against strong non-neural methods

Incorrectly implemented baselines [4, 6]
E.g., different implementations of the same method can vary in performance

[1] Ferrari Dacrema, Maurizio, et al. A troubling analysis of reproducibility and progress in recommender systems research. In TOIS 2021.
[2] Shehzad, Faisal, and Dietmar Jannach. Everyone's a winner! On hyperparameter tuning of recommendation models. In RecSys 2023.
[3] Li, Ming, et al. A next basket recommendation reality check. In TOIS 2023.
[4] Petrov, Aleksandr, and Craig Macdonald. A systematic review and replicability study of bert4rec for sequential recommendation. In RecSys 2022.
[5] Lin, Jimmy. The neural hype and comparisons against weak baselines. In SIGIR Forum 2019.
[6] Qin, Zhen, et al. Are neural rankers still outperformed by gradient boosted decision trees?. In ICLR 2021.

Untuned baselines

Shehzad and Jannach [1] surveyed 21 recommender systems from KDD, RecSys, SIGIR, TheWebConf, and WSDM in 2022 and found:

- 6/21 papers contain **no information** about hyperparameters at all.
- 4/21 papers **copy parameters** from previous work.
- 4/21 papers use the **same parameters across datasets.**
- 7/21 papers list **parameter ranges** but **not the tuning method**.

Only two papers describe **parameter ranges, the final values, tuning methods, and tune across datasets**.

Only one of the two papers also released their code.

[1] Shehzad, Faisal, and Dietmar Jannach. Everyone's a winner! On hyperparameter tuning of recommendation models. In RecSys 2023.

Untuned baselines

The authors go on to demonstrate **the importance of tuning baselines:**

Even the worst-performing tuned model outperformed all other untuned methods!

In short, **everyone is a winner!**

Tuned models		
ML-1M		
Model	*nDCG@10*	*Model*
Mult-DAE	0,300	NeuMF
Mult-VAE	0,294	Mult-VAE
GMF	0,280	GMF
NeuMF	0,277	Mult-DAE
ONCF	0,225	*MostPop*
MostPop	*0,162*	ConvMF
ConvMF	0,160	NGCF
NGCF	0,100	ONCF
Non-tuned models	>	
Mult-DAE	0,071	Mult-DAE
ONCF	0,037	Mult-VAE
ConvMF	0,022	ConvMF
NeuMF	0,021	GMF
GMF	0,016	NGCF
NGCF	0,013	ONCF
Mult-VAE	0,006	NeuMF

Comparison of tuned and untuned models on ML-1M [1]

[1] Shehzad, Faisal, and Dietmar Jannach. Everyone's a winner! On hyperparameter tuning of recommendation models. In RecSys 2023.

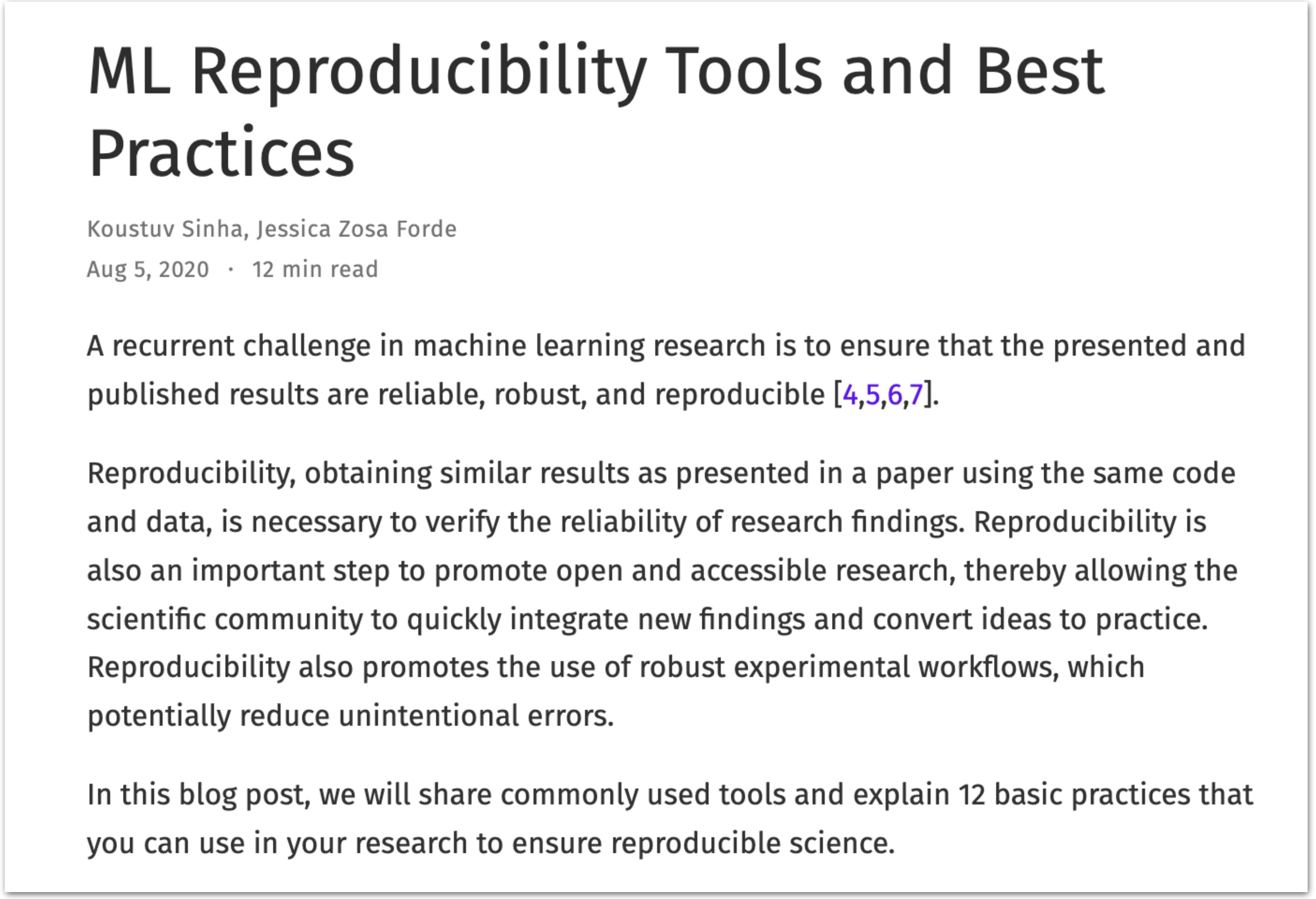
Making reproducibility easier

Some opinionated tips and tools that might be useful

Useful Tools and Libraries

- Dependency management
- Config management
- Parameter tuning
- Managing experiments
- Data management
- Documentation

…



ML Reproducibility Tools and Best Practices

Koustuv Sinha, Jessica Zosa Forde

Aug 5, 2020 · 12 min read

A recurrent challenge in machine learning research is to ensure that the presented and published results are reliable, robust, and reproducible [4,5,6,7].

Reproducibility, obtaining similar results as presented in a paper using the same code and data, is necessary to verify the reliability of research findings. Reproducibility is also an important step to promote open and accessible research, thereby allowing the scientific community to quickly integrate new findings and convert ideas to practice. Reproducibility also promotes the use of robust experimental workflows, which potentially reduce unintentional errors.

In this blog post, we will share commonly used tools and explain 12 basic practices that you can use in your research to ensure reproducible science.

An overview from the organizers of the ML Reproducibility Challenge (MLRC) [1]

[1] Koustuv Sinha, Robert Stojnic - ML Reproducibility Tools and Best Practices: https://koustuvsinha.com//practices_for_reproducibility/

Dependency management using UV

UV is a fast package & project manager, I'd recommend it over Mamba, Conda, or Poetry:

Features [1]:

- Extremely fast dependency resolution
- Lockfile and pyproject.toml support
- Ruff linter & formatter (with Jupyter support)
- Easily publish packages

```
# Create virtual environments:
uv venv my_env --python 3.13

# Install dependencies:
uv add numpy

# Lock dependency versions:
uv lock

# Run build tools, e.g., the ruff linter:
uvx ruff check

# Ruff formatter:
uvx ruff format
```

[1] UV package manager: https://docs.astral.sh/uv/getting-started/features/
[2] Ruff: https://docs.astral.sh/ruff/formatter/

Config management using Hydra

Replace argparse with config files [1]:

```
python train.py \
    --optimizer adam \
    --learning_rate 0.0003 \
    --weight_decay 0.9 \
    --model model_a \
    --layers 5 \
    --hidden_dim 100 \
    --dropout 0.2 \
    --activation gelu \
...
```

→

```
# config.yaml
optimizer: adam
learning_rate: 0.0003
weight_decay: 0.9

# model/model_a.yaml
layers: 5
hidden_dim: 100
activation: gelu
dropout: 0.2

# model/model_b.yaml
layers: 3
hidden_dim: 512
activation: elu
```

[1] Hydra config manager: https://hydra.cc/

Config management using Hydra

1. Compose & override configurations:

```
python train.py model=model_a data=data_b
```

2. Sweep hyperparameter configurations and ranges

```
python train.py model=model_a,model_b learning_rate=0.01,0.001,0.0001
```

3. Run in parallel on SLURM

```
python train.py model=model_a,model_b +hydra/launcher=submitit_slurm
```

Instantiate objects, assemble experiments, search hyperparameters, logging, ...

Be aware of too complex parameter configurations, it makes code hard to follow.

[1] Hydra config manager: https://hydra.cc/

Tuning hyperparameters

Google's Deep Learning Tuning Playbook [1]

- Start simple and make **incremental improvements.**
- First, **explore your parameter space** through random or grid search.
- Learn about scientific, nuisance, and fixed hyperparameters to know what to tune each round.
- **Maximize performance** with black-box optimizers only when you understand your parameters well (e.g., using Optuna, Nevergrad, or Ax).

```
python train.py \
    layers=2,3,4 \ # Scientific
    learning_rate=0.03,0.003,0.0003 \ # Nuisance
    dropout=0.25 # Fixed
```

To tune model depth (scientific param), we always need to adjust the learning rate (nuisance param):

[1] https://github.com/google-research/tuning_playbook?tab=readme-ov-file#a-scientific-approach-to-improving-model-performance

Useful Tools and Libraries

- Dependency management
- Config management
- Parameter tuning
- **Managing experiments**
- Data management
- Documentation

…

Many tools can make experimentation easier, e.g. [1]:

- Track experiments (names, parameters, versions, etc.)
- Plot metrics in real-time
- Checkpoint models and data artifacts
- Integrate hyperparameter tuning libraries
- Share results with collaborators

Tools: Weights & Biases, MLFlow, Comet.ML, Neptune.ai, Aim, TensorBoard, PyTorch Lightning

[1] Koustuv Sinha, Robert Stojnic - ML Reproducibility Tools and Best Practices: https://koustuvsinha.com//practices_for_reproducibility/

Useful Tools and Libraries

- Dependency management
- Config management
- Parameter tuning
- Managing experiments
- **Data management**
- Documentation

...

Use established libraries in your field:
HuggingFace (HF) datasets

Publish your datasets in permanent locations:

- Your institution?
- HF datasets (up to 300GB), DVC (unlimited)

Document your datasets:
Datasheets [2], HF dataset cards, Google data cards

[1] MacAvaney, Sean, et al. Simplified data wrangling with ir_datasets. In SIGIR 2021.
[2] Gebru, Timnit, et al. Datasheets for datasets. Communications of the ACM 2021.

Useful Tools and Libraries

- Dependency management
- Config management
- Parameter tuning
- Managing experiments
- Data management
- **Documentation**

  …

Document your model [1]:

- Authors, license, funding
- Model architecture, training, evaluation
- Risks, limitations, biases
- Carbon emissions [3]
- Usage examples
- Citation

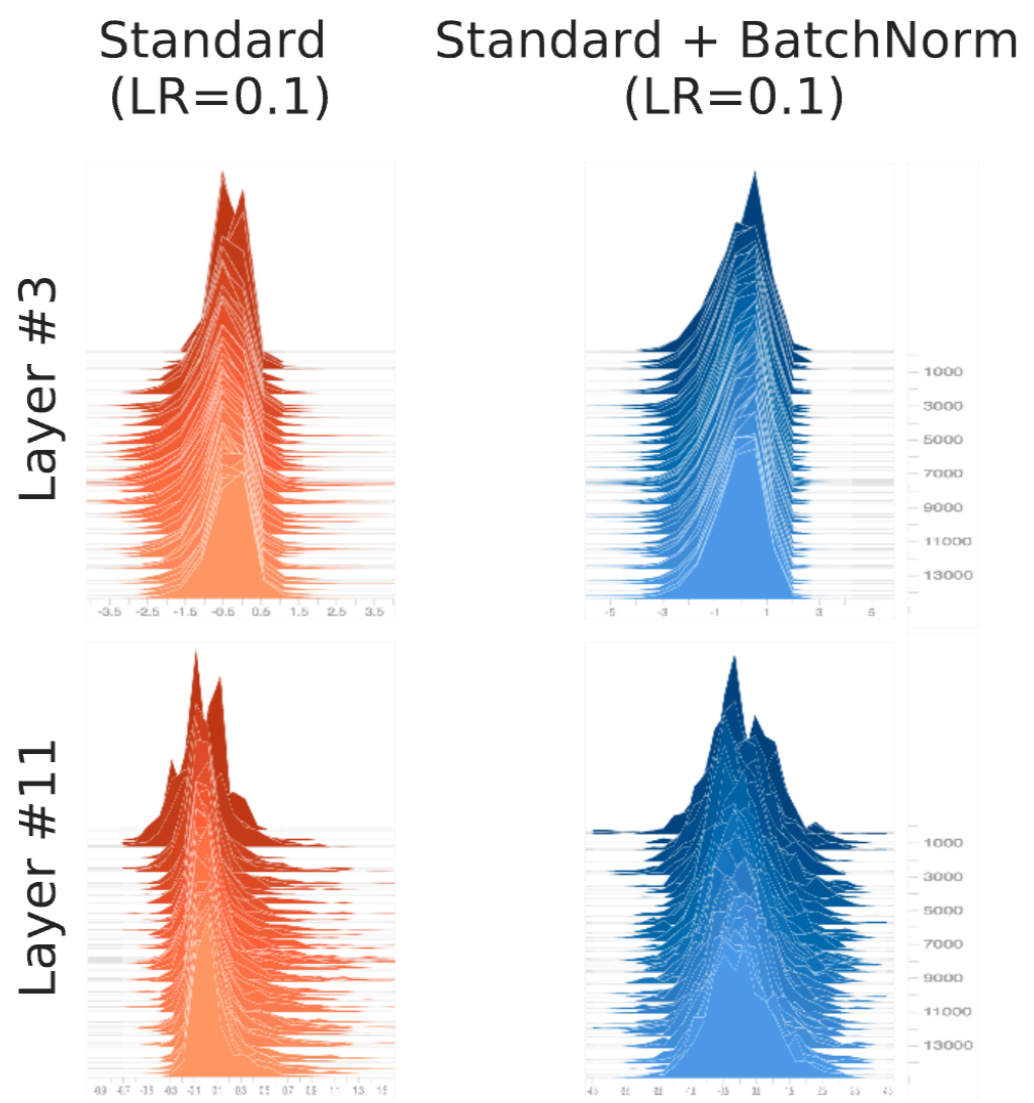
See [2] for a comprehensive overview of documentation tools.

[1] Mitchell, Margaret, et al. Model cards for model reporting. In FAccT 2019.
[2] https://huggingface.co/docs/hub/model-card-landscape-analysis#summary-of-ml-documentation-tools
[3] https://mlco2.github.io/impact/

A word on alchemy

Why does BatchNorm work?

- BatchNorm was a highly successful innovation, allowing to train much deeper NNs.
- The original paper stated that it reduces "Internal Covariate Shift" of activations, without ever demonstrating the problem or rigoroulsy defining why it speeds up SGD.
- Rahimi publicly criticized our lack of understanding in DL research in 2017 as Alchemy [3].
- Later, Santurkar et al. [2] found no pronounced "covariate shift", but rather a smoothing of the loss landscape (smaller gradients).



Input distributions showing little covariate shift over time [2].

[1] Ioffe, Sergey, and Christian Szegedy. "Batch normalization: Accelerating deep network training by reducing internal covariate shift." In ICML 2015.
[2] Santurkar, Shibani, et al. "How does batch normalization help optimization?." In NeurIPS 2018.
[3] Ali Rahimi's test of time award speech at NeurIPS 2017. https://www.youtube.com/watch?v=Qi1Yry33TQE

Exploratory vs. Empirical Research

Bouthillier et al. see BatchNorm as **exploratory research [1]:**

- BatchNorm was very impactful (as the observations generalized).
- But it took more rigorous empirical follow-up work to explain it.
- However, important follow-up work on normalization was published [2, 3] before the Internal Covariate Shift hypothesis was debunked.

Bouthillier et al. argue that both **exploratory** and **empirical research** are important, the balance is important.

In the alchemy debate, Yann LeCun warned of the streetlight effect [4,5].

[1] Bouthillier, Xavier, César Laurent, and Pascal Vincent. "Unreproducible research is reproducible." In ICML 2019.
[2] Salimans, Tim, and Durk P. Kingma. "Weight normalization: A simple reparameterization to accelerate training of deep neural networks." In NeurIPS 2016.
[3] Ba, Jimmy Lei, Jamie Ryan Kiros, and Geoffrey E. Hinton. "Layer normalization." arXiv preprint arXiv:1607.06450 (2016).
[4] Yann LeCun, Answer to Ali Rahimi's test of time award in 2017: https://www.facebook.com/yann.lecun/posts/10154938130592143
[5] Visualization from https://sketchplanations.com/looking-under-the-lamppost

Writing reproducibility papers

Those in glass houses ...

A reproducibility paper should be reproducible [1, 2, 3]:

> [–] **Reproduction hinting add valid flaws, but repeats similar mistakes**
> *ML Reproducibility Challenge 2022*

A review for a reproducibility paper at MLRC 2022.

Not including all code/data in a reproducibility paper is a **reason for desk rejection** at some conferences [2].

[1] SIGIR 24: https://sigir-2024.github.io/call_for_res_rep_papers.html
[2] RecSys 24: https://recsys.acm.org/recsys24/call/#content-tab-1-1-tab
[3] ECIR24: https://www.ecir2024.org/2023/07/10/call-for-reproducibility-papers/

New and important lessons

*"We are particularly interested in **reproducibility papers** (different team, different experimental setup) **rather than replicability papers** (different team, same experimental setup). The emphasis is [...] on generating new research insights with existing approaches [1]."*

Key points to consider [1, 2, 3, 4]:

- **Novelty:** Are your findings and your setup novel?
- **Generalizability:** Which lessons from prior work hold up?
- **Impact:** Are your conclusions important for the scientific community?

[1] SIGIR 24: https://sigir-2024.github.io/call_for_res_rep_papers.html
[2] RecSys 24: https://recsys.acm.org/recsys24/call/#content-tab-1-1-tab
[3] ECIR24: https://www.ecir2024.org/2023/07/10/call-for-reproducibility-papers/
[4] MLRC 23: https://reproml.org/call_for_papers/

Everybody makes mistakes

Involve the original authors in the process
Ask for code, ask questions, discuss findings, send the final manuscript, and plan adequate response times (e.g., 30 days).

The golden rule
Write the paper as if somebody else writes about your work.

Hanlon's razor [1]
Never attribute to malice that which can be adequately explained by neglect, ignorance, or incompetence.*

[1] Arthur Bloch. Murphy's Law Book Two: More Reasons Why Things Go Wrong! p. 52. ISBN 9780417064505, 1980.
* the original quote just states: "[...] *adequately explained by stupidity*", but I think the version above is more useful.

Conclusion

Concluding

- Reproducibility is at the **heart of scientific progress in ML research.**
- Producing truly reproducible work is much **more than just publishing code.**
- Select **strong baseline implementations** and **tune them with care.**
- Tools can make reproducibility easier, but ultimately, it comes down to continually striving to **publish clear, open, and detailed research in exchange with our peers.**
- When conducting reproducibility work, focus on **novelty, generalizability, and impact** of an idea, and try to **involve the original authors.**

Thank you for listening!

Any questions?

Day and Night - M.C. Escher, 1938