repro_eval: A Python Interface to Reproducibility Measures of System-oriented IR Experiments

Timo Breuer[1], Nicola Ferro[2], Maria Maistro[3], and Philipp Schaer[1]

[1] TH Köln - University of Applied Sciences, Germany
{timo.breuer,philipp.schaer}@th-koeln.de
[2] University of Padua, Italy
ferro@dei.unipd.it
[3] University of Copenhagen, Denmark
mm@di.ku.dk

Abstract. In this work we introduce repro_eval - a tool for reactive reproducibility studies of system-oriented Information Retrieval (IR) experiments. The corresponding Python package provides IR researchers with measures for different levels of reproduction when evaluating their systems' outputs. By offering an easily extensible interface, we hope to stimulate common practices when conducting a reproducibility study of system-oriented IR experiments.

Keywords: Replicability · Reproducibility · Evaluation.

1 Introduction

Reproduciblity is a cornerstone of scientific findings. However, many scientific fields are affected by reproducibility issues [2] and IR is not an exception [6]. In the previous decade, different communities from the computational sciences developed a range of tools supporting researchers in their attempts to make studies reproducible.

According to Potthast et al. [12] reproducibility efforts can be subdivided into either *proactive*, *reactive* or *supportive* actions. Many exisiting tools for reproducibility support *proactive* actions. More general examples include RoHub [11], CodaLab[4] (executable papers), ReproZip [4] (workflow tracking, data provenance), Process Migration Framework (system resource logging) [13], ReproMatch[5] (search engine for reproducibility tools), noWorkflow [10] (monitoring data provenance), yesWorkflow [9] and others. With special regards to system-oriented IR experiments, the implementations and requirements can be *proactively* packaged with virtual machines or as shown more recently with Docker

[4] https://codalab.org/

[5] http://repromatch.poly.edu/tools/search/

containers as exemplified by TIRA [12] and the OSIRRC platform [5], respectively. On the other hand, the IR community promotes *reactive* reproducibility studies by archiving experimental data from evaluation campaigns at TREC [15] or CLEF [1]. Here, we can use the artifacts - or more specifically system runs - of previous experiments as points of reference to which we compare the results of our reimplementations. Tools of *supportive* actions have been realized as *Evaluation-as-a-Service* infrastructures and shared task platforms [8].

The presented software complements existing reproducibility tools by measuring the exactness of reproduced system runs in relation to their original counterparts. It is often not sufficient to compare system results based on their average retrieval performance (ARP), as the averaged scores may hide differences between the distributions of topic scores or the order of documents. In this sense, `repro_eval` supports researchers as part of their *reactive* approach when reimplementing another researcher's retrieval system. The implemented measures of `repro_eval` provide the reproducer with insights at different levels of reproduction. Under consideration of these insights, `repro_eval` contributes to the adequate use of reimplemented systems, for instance when they are used as baseline systems in experimental evaluations.

2 Evaluating Reimplementations with `repro_eval`

The presented Python package compiles system-oriented reproducibility measures we introduced in previous studies [3]. According to the ACM policy of *Artifact Review and Badging*[6], we align the system-oriented IR experiment to the terminology it introduces. More specifically, `repro_eval` can be used to evaluate the *reproducibility* with a reimplemented IR system in combination with the *same test collection* of the original experiments, whereas *replicability* considers the reimplementation in combination with a *different test collection*.

In this sense, `repro_eval` supports IR researchers who want to compare their systems to a reference or state-of-the-art system for which no source code or public artifact is available. Especially, when reference systems need to be evaluated in a different context (with a possibly different test collection), IR researchers cannot rely on the results reported in the original publication. With `repro_eval` they can evaluate their reimplemented reference system and gain insight into how similar the two systems are. With an increasing level of specifity, the Python package provides different measures that provide a more nuanced perspective on the degree of reproduction and replication. Figure 1 provides a hierarchical illustration of the different levels and corresponding measures.

Proceeding from the bottom to the top of this hierarchy, the specifity of reproduction (and replication) increases from the most general to the most specific. Note that some evaluations are limited to reproduced experiments only. The ordering of documents can only be compared if all systems runs (possibly) contain the same documents or were derived from the same test collection.

[6] https://www.acm.org/publications/policies/artifact-review-badging Previous versions of the policy basically swapped the meaning of the two terms *reproducibility* and *replicability*, which is why we used the terms vice versa in earlier studies.

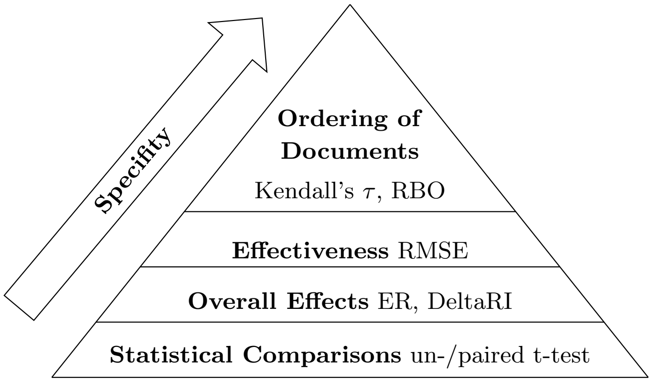


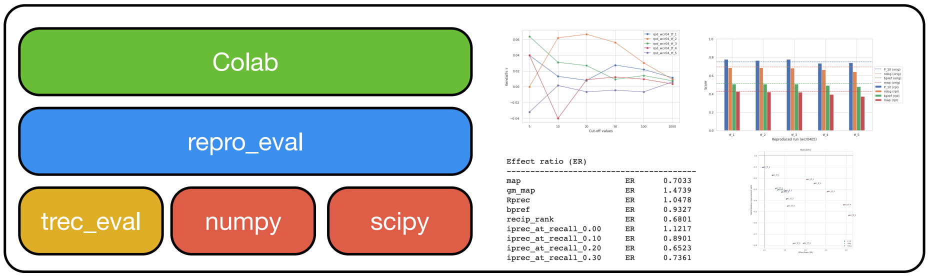
Fig. 1: Measures of `repro_eval` arranged with regard to their level of specifity

Likewise, the level of effectiveness can only be determined if reproduced runs are derived for the same topics as in the original experiment. Here, the Root Mean Square Error (RMSE) evaluates the closeness of the topic scores distributions between the reproduced and original results. In order to evaluate replicated runs, reimplementations need to be compared on more general levels. The overall effects are determined with the help of the Effect Ratio (ER) and the Delta Relative Improvement (DeltaRI). To do so, a replicated baseline run and an improved version of it (which we refer to as the advanced run) are required. The ER and DeltaRI measure how accurately the effects between the baseline and the advanced run can be replicated. At the most general level, it is possible to compare the topic score distributions of the reproduced and replicated runs with paired and unpaired t-tests, respectively. The p-values deliver information about the success of reproduction and replication. In case of a low p-value, there is a strong evidence that the repeated experiment has failed.

3 Case Study on the Evaluation of Reproducibility

Let us consider IR researchers reimplementing a retrieval system of another research group that provides no other artifacts except for the description in the publication and the original run files. Having reimplemented the system, the researchers want to know about the quality of their reproductions/replications. Since the publication lacks some details about optional processing steps or parameterizations, the researchers try different variations and end up having many runs. How do they know which one is the most exactly reproduced/replicated run? Intuitively, they can compare the runs by the ARP. However, equal (averaged) scores might hide differences between the topic score distributions or document orderings. Furthermore, replicated runs (derived from another test collection) cannot be compared at these two levels.

In this case, `repro_eval` provides a toolbox of different measures for reproducibility and replicability. It is a Python package which uses the `Pytrec_eval` [7]



```
Effect ratio (ER)
----------------------------------------
map                          ER      0.7033
gm_map                       ER      1.4739
Rprec                        ER      1.0478
bpref                        ER      0.9327
recip_rank                   ER      0.6801
iprec_at_recall_0.00         ER      1.1217
iprec_at_recall_0.10         ER      0.8901
iprec_at_recall_0.20         ER      0.6523
iprec_at_recall_0.30         ER      0.7361
```

Fig. 2: `repro_eval` as a cornerstone for statistical and visual analytics of reproducibility studies with the help of Colab.

interface to `trec_eval`[7], as well as `numpy` [16] and `scipy` [14]. Once installed, `repro_eval` can be used either by a conventional command line call (similar to `trec_eval`) or by importing it into programs as exemplified by the Colab-based tool for visually analysing the reproducibilty and replicability (see Figure 2).

We provide an interactive demonstration in a Colab-based environment featuring example data that complies with the previously outlined use case[8]. Besides numerical outputs comparable to those of `trec_eval`, our demonstration showcases some plots that help researchers to gain a better understanding of the reproductions. Bar plots visualize conventional comparisons at the level of ARP, whereas the included plots of Kendall's τ Union and the RMSE illustrate the reproduction quality across the cut-off ranks. At the level of overall effects, the ER/DeltaRI plots are a valuable tool helping to explore the space of reproduction/replication. In theory, the best reproduction/replication yields (ER 1 / DeltaRI 0). The included scatter plots visualize which runs resemble the originals in terms of P@10, AP, and nDCG the most.

4 Contributions & Conclusions

We introduce `repro_eval`, a tool for reproducibility studies of system-oriented IR experiments. This tool provides a Python package that can be used by researchers in their reactive approach to reimplement another researchers' experiments. The included reproducibility and replicability measures offer assistance when measuring the closeness of reimplemented systems' outputs compared to the original results. More technical details, installation instructions and a demonstration video of `repro_eval` can be found in our public GitHub repository[9].

Acknowledgements This paper was partially supported by the EU Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No. 893667, and by the German Research Foundation (No. 407518790).

[7] https://github.com/usnistgov/trec_eval

[8] https://colab.research.google.com/github/irgroup/repro_eval/blob/master/example/demo.ipynb

[9] https://github.com/irgroup/repro_eval

References

1. Agosti, M., Nunzio, G.M.D., Ferro, N., Silvello, G.: An innovative approach to data management and curation of experimental data generated through IR test collections. In: Ferro, N., Peters, C. (eds.) Information Retrieval Evaluation in a Changing World - Lessons Learned from 20 Years of CLEF, The Information Retrieval Series, vol. 41, pp. 105–122. Springer (2019). https://doi.org/10.1007/978-3-030-22948-1_4, https://doi.org/10.1007/978-3-030-22948-1_4
2. Baker, M.: 1,500 scientists lift the lid on reproducibility. Nature **533**, 452–454 (2016)
3. Breuer, T., Ferro, N., Fuhr, N., Maistro, M., Sakai, T., Schaer, P., Soboroff, I.: How to measure the reproducibility of system-oriented IR experiments. In: Huang, J., Chang, Y., Cheng, X., Kamps, J., Murdock, V., Wen, J., Liu, Y. (eds.) Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, SIGIR 2020, Virtual Event, China, July 25-30, 2020. pp. 349–358. ACM (2020). https://doi.org/10.1145/3397271.3401036, https://doi.org/10.1145/3397271.3401036
4. Chirigati, F., Rampin, R., Shasha, D.E., Freire, J.: Reprozip: Computational reproducibility with ease. In: Özcan, F., Koutrika, G., Madden, S. (eds.) Proceedings of the 2016 International Conference on Management of Data, SIGMOD Conference 2016, San Francisco, CA, USA, June 26 - July 01, 2016. pp. 2085–2088. ACM (2016). https://doi.org/10.1145/2882903.2899401, https://doi.org/10.1145/2882903.2899401
5. Clancy, R., Ferro, N., Hauff, C., Lin, J., Sakai, T., Wu, Z.Z.: The SIGIR 2019 open-source IR replicability challenge (OSIRRC 2019). In: Piwowarski, B., Chevalier, M., Gaussier, É., Maarek, Y., Nie, J., Scholer, F. (eds.) Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2019, Paris, France, July 21-25, 2019. pp. 1432–1434. ACM (2019). https://doi.org/10.1145/3331184.3331647, https://doi.org/10.1145/3331184.3331647
6. Ferro, N.: Reproducibility challenges in information retrieval evaluation. J. Data and Information Quality **8**(2), 8:1–8:4 (2017). https://doi.org/10.1145/3020206, https://doi.org/10.1145/3020206
7. Gysel, C.V., de Rijke, M.: Pytrec_eval: An extremely fast python interface to trec_eval. In: Collins-Thompson, K., Mei, Q., Davison, B.D., Liu, Y., Yilmaz, E. (eds.) The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, SIGIR 2018, Ann Arbor, MI, USA, July 08-12, 2018. pp. 873–876. ACM (2018). https://doi.org/10.1145/3209978.3210065, https://doi.org/10.1145/3209978.3210065
8. Hopfgartner, F., Hanbury, A., Müller, H., Eggel, I., Balog, K., Brodt, T., Cormack, G.V., Lin, J., Kalpathy-Cramer, J., Kando, N., Kato, M.P., Krithara, A., Gollub, T., Potthast, M., Viegas, E., Mercer, S.: Evaluation-as-a-service for the computational sciences: Overview and outlook. ACM J. Data Inf. Qual. **10**(4), 15:1–15:32 (2018). https://doi.org/10.1145/3239570, https://doi.org/10.1145/3239570
9. McPhillips, T.M., Song, T., Kolisnik, T., Aulenbach, S., Belhajjame, K., Bocinsky, K., Cao, Y., Chirigati, F., Dey, S.C., Freire, J., Huntzinger, D.N., Jones, C., Koop, D., Missier, P., Schildhauer, M., Schwalm, C.R., Wei, Y., Cheney, J., Bieda, M., Ludäscher, B.: Yesworkflow: A user-oriented, language-independent tool for recovering workflow information from scripts. CoRR **abs/1502.02403** (2015), http://arxiv.org/abs/1502.02403

10. Murta, L., Braganholo, V., Chirigati, F., Koop, D., Freire, J.: noworkflow: Capturing and analyzing provenance of scripts. In: Ludäscher, B., Plale, B. (eds.) Provenance and Annotation of Data and Processes - 5th International Provenance and Annotation Workshop, IPAW 2014, Cologne, Germany, June 9-13, 2014. Revised Selected Papers. Lecture Notes in Computer Science, vol. 8628, pp. 71–83. Springer (2014). https://doi.org/10.1007/978-3-319-16462-5_6, https://doi.org/10.1007/978-3-319-16462-5_6
11. Palma, R., Holubowicz, P., Corcho, Ó., Gómez-Pérez, J.M., Mazurek, C.: Rohub - A digital library of research objects supporting scientists towards reproducible science. In: Presutti, V., Stankovic, M., Cambria, E., Cantador, I., Iorio, A.D., Noia, T.D., Lange, C., Recupero, D.R., Tordai, A. (eds.) Semantic Web Evaluation Challenge - SemWebEval 2014 at ESWC 2014, Anissaras, Crete, Greece, May 25-29, 2014, Revised Selected Papers. Communications in Computer and Information Science, vol. 475, pp. 77–82. Springer (2014). https://doi.org/10.1007/978-3-319-12024-9_9, https://doi.org/10.1007/978-3-319-12024-9_9
12. Potthast, M., Gollub, T., Wiegmann, M., Stein, B.: TIRA integrated research architecture. In: Ferro, N., Peters, C. (eds.) Information Retrieval Evaluation in a Changing World - Lessons Learned from 20 Years of CLEF, The Information Retrieval Series, vol. 41, pp. 123–160. Springer (2019). https://doi.org/10.1007/978-3-030-22948-1_5, https://doi.org/10.1007/978-3-030-22948-1_5
13. Rauber, A., Miksa, T., Mayer, R., Pröll, S.: Repeatability and re-usability in scientific processes: Process context, data identification and verification. In: Kalinichenko, L.A., Starkov, S. (eds.) Selected Papers of the XVII International Conference on Data Analytics and Management in Data Intensive Domains (DAMDID/RCDL 2015), Obninsk, Russia, October 13-16, 2015. CEUR Workshop Proceedings, vol. 1536, pp. 246–256. CEUR-WS.org (2015), http://ceur-ws.org/Vol-1536/paper33.pdf
14. Virtanen, P., Gommers, R., Oliphant, T.E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., van der Walt, S., Brett, M., Wilson, J., Millman, K.J., Mayorov, N., Nelson, A.R.J., Jones, E., Kern, R., Larson, E., Carey, C.J., Polat, I., Feng, Y., Moore, E.W., VanderPlas, J., Laxalde, D., Perktold, J., Cimrman, R., Henriksen, I., Quintero, E.A., Harris, C.R., Archibald, A.M., Ribeiro, A.H., Pedregosa, F., van Mulbregt, P., SciPy: Scipy 1.0-fundamental algorithms for scientific computing in python. CoRR **abs/1907.10121** (2019), http://arxiv.org/abs/1907.10121
15. Voorhees, E.M., Rajput, S., Soboroff, I.: Promoting repeatability through open runs. In: Yilmaz, E., Clarke, C.L.A. (eds.) Proceedings of the Seventh International Workshop on Evaluating Information Access, EVIA 2016, a Satellite Workshop of the NTCIR-12 Conference, National Center of Sciences, Tokyo, Japan, june 7, 2016. National Institute of Informatics (NII) (2016), http://research.nii.ac.jp/ntcir/workshop/OnlineProceedings12/pdf/evia/04-EVIA2016-VoorheesE.pdf
16. van der Walt, S., Colbert, S.C., Varoquaux, G.: The numpy array: A structure for efficient numerical computation. Comput. Sci. Eng. **13**(2), 22–30 (2011). https://doi.org/10.1109/MCSE.2011.37, https://doi.org/10.1109/MCSE.2011.37